

SINCLAIR

Volume 2, Issue 12 £2.50

6 7 8 14 15 19 21 27
QL

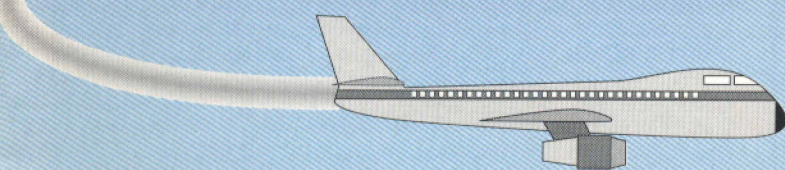
WORLD

**Machine Code
for Beginners**
TRAP CALLS

**Reach
for the**

SKY!

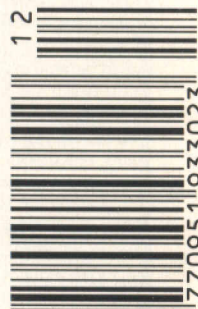
Airliner flightdeck for the QL



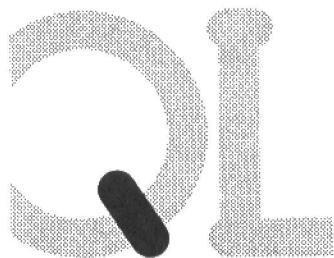
THE ULTIMATE SQUEAK
The DIY Mouse adapted to other programs.

ISSN 0951-9335

12 >



9 770951 933023



VOL 2 ISSUE 12 Contents

Editor

Helen Armstrong

Publisher

Mark Kasprovicz

Advertising Manager

Jim Peskett

Creative Director

John Stanley

Graphic Artist

Stevie Billington

Magazine Services

Linda Miller, Frances Maxwell,
Pauline Wakeling

Sinclair QL World,
Published by Arcwind Ltd.
The Blue Barn,
Tew Lane, Wootton,
Woodstock,
Oxon. OX7 1HA
Tel: 0993 811181
Fax: 0993 811481
ISSN 026806X

If you have any comments or difficulties please write to the editor and we will do our best to deal with your problem in the magazine, though we cannot guarantee individual replies.

Back issues are available from the publisher price £2.50 UK, £2.99 Europe. Overseas rates on request.

Subscriptions from: Arcwind
The Blue Barn, Tew Lane,
Wootton, Woodstock, Oxon.
OX7 1HA
UK: £23.40
Europe: £32.90
Rest of World: £40.90

Reprographic Services:
Eclipse, Brook Street,
Watlington, Oxon. OX9 5JH.
Distributed by: Seymour Press
Ltd., Windsor House, 1270
London Road, Norbury,
London, SW16 4DH

© 1994 ARCWIND LTD.
Sinclair QL World is published 12 times a year. All rights reserved. Reproduction in whole or in part without written permission is strictly prohibited. We welcome contributions. All material must be supplied with a SAE.

While all reasonable care is taken in compiling QL World, the publisher and its agents assume no responsibility in affects arising therefrom. Opinions expressed are those of the authors.

6 QL SCENE

Disk interfaces from Qubbesoft ... Multitasking chess from Merz ... Digital Precision's 10 years ... Front end from Hessler ... International Meeting in Germany in February ... TFServices I2C bus interfaces.

8 OPEN CHANNEL

Changing hex bytes ... downloadable fonts confusion ... Franz Herrmann calling from Germany ...

10 REVIEW: FLIGHTDECK

Ray Dawson flies the latest upgrade on Deltasoft's flight simulator.

14 USER REPORT: KEYBOARD 90 INTERFACE

Alex Munden with Falkenburg's extension keyboard kit.

15 THE NEW USER GUIDE - part 31

Concepts: five functional layers between the computer and the user.

19 ITALIA

Simon Goodwin goes to the show at Lake Como.

21 DIY TOOLKIT

Simon Goodwin gets the mouse to emulate keyboard entry.

27 MACHINE CODE FOR BEGINNERS - Part 7

Alan Bridewell introduces Trap Calls.

31 MICRO ADS**Coming Soon**

QLCalendar - HDD disk and new QL boxing system - WorldMap.

QLScene

Front End From Hessler

Albin Hessler has produced a new desktop front end for Qdos-compatible systems. Billed as "the ultimate pointer-driven desktop program for Qdos-compatible systems", **CueShell** has freely-sizable windows, drag-and-drop graphics copying, pan/scroll/slide bars, system/jobs/hotkeys control, saving of window sizes and catalogue sort orders, supports all screen resolutions and is easy to configure and put into use. (Albin Hessler has specialised in making processes easier in the past.)

CueShell can be used for file management, or for file-name selection from other programs. It requires expanded memory, and Albin recommends use of a mouse. Upgrades will be available for the cost of postage in International Reply Coupons.

The program will cost about DM100. Enquiries to **Albin Hessler Software, Im Zeilfeld 25, D-72631 Aichtal, Germany. Tel (and fax) 010 49 7127 56280.**

TF Services takes the I2C Bus

Tony Firshman of **TF Services** has released the first two in a new family of I2C bus **hardware interfaces** - here is the information we promised you.

The I2C bus was designed by the Dutch electronics giant Philips to simplify hardware interfacing to computers. The Minerva MKII (RTC) rom adds a battery-backed ram and a clock using the I2C bus. Computer users can also get a cable and a 9-pin D connector to allow other I2C devices to be added. There are a wide range of interesting items from phone diallers, LCD and relay drivers, teletext chips to microprocessors and many radio and TV controllers. The Minerva will not, however, allow the use of microprocessors and other devices needing dual bus masters with the QL.

TF Services' interfaces are based on the parallel and analogue I2C chips.

The interfaces themselves are in small boxes with high quality double-sided circuit boards. They are linked to each other by 15-pin cable-less D connectors, and each interface contains two I2C chips, each of which can have one of 8 addresses (set by DIP switches). Power and signal lines are taken from the Minerva connector, and at least four interfaces can be powered without using an external power supply.

The software which comes with the connectors is designed to make reading and writing to the devices easy. Only the device address (0 to 7) is needed to read from either device, and to write, only the address and data are needed. The code can multitask in compiled SuperBasic programs. Machine code traps are provided for machine code programmers. Both devices are accessed at close to the maximum speed to 100 Kbits per second.

Data is only received and sent on a single data line after successful completion of a handshake protocol, which includes a unique I2C chip address.

The **ANALOGUE INTERFACE** can handle 8 bytes ADC (analogue to digital) and two bytes DAC (output). It can be used directly for such tasks as frequency analysing - one customer who works for tv transmission links is planning to use the QL and an analogue interface to link to an x/y frequency plotter. Other uses include temperature measurement - taking direct readings off many temperature probes, with only a resistor network added to give the appropriate voltage range. TF Services use rheostats to provide position readings when running their "intelligent" robot demonstrations. Simon Goodwin is investigating sound sampling and playback. (Because of speed on the I2C bus, music will never be very high quality, but acceptable speech should be possible. It could be used as the basis of a computer-based multimeter, low frequency meter/low frequency oscilloscope.

The **PARALLEL INTERFACE** provides 16 two-way lines to read binary data (high/low). Output is high enough to drive LEDs, but has to be stepped up to drive relays/motors and so on. It can also be used for direct input to many digital model train controllers.

Future parallel circuit developments include a "power driver" to switch up to four amps. This is mainly to drive up to eight motors unidirectionally, but can also link into an 8-relay device (also on the drawing board) for mains or low-voltage switching.

The Parallel interface is priced £25, and the analogue interface £30, from **TF Services, Holly Corner, Priory Road, Chavey Down, Ascot, Berks SL5 8RL**. Note TF's new address! **Tel. 0344 890986. Fax and scrolling modem 0344 890987.**

I2C data sheets, and control software and manuals can be bought for £2 an item without the interfaces.

International Meeting in February

Sinclair QL User Club eV are to hold an International QL Meeting in Germany on Saturday 19th February 1994.

The meeting will be held at the University of Bielefeld in North East Germany, not very far from Munster where International QL Meetings have been held before.

Franz Herrmann has prepared a full pack of information including street maps, train connections, accommodation and sightseeing.

The address to contact is **Sinclair QL User Club eV, c/o Franz Herrmann, Talstrasse 21, D-53545 Ockenfels, Germany, or via Internet: Franz.Herrmann@bn.maus.de.** (No unencoded material, no large texts please.)

See Franz's letter in **Open Channel** this month. Also reported to be happening some time next year (no dates fixed yet) are meetings arranged by QItaly and Ergon Development in Italy, and IQLR in Newport, USA.

QUBBESOFT GO INTO DISK INTERFACES

Qubbesoft now expect to release their **Fastnet** QL networking interface at the end of January, priced around £120 for a two-board unit, which will link QL to QL, ST to ST or QL to ST. The boards are with the pcb makers as we write.

Ron will be showing off Fastnet at Andrew Frannick's QL show at Blackrod, Bolton on 23rd January; Quanta's Edinburgh show on 12th February, and Franz Herrman's International QL Meeting in Bielefeld, Germany, on 19th February.

Qubbesoft are also launching a new QL hardware product shortly following the Fastnet. This is an **IDE hard disk interface, codenamed QUBIDE**. "Stuart thought up the name", says Ron. Miracle's Stuart Honeyball also thought of a way to connect the interface to the QL rom slot to plug-and-go, without an extra adaptor board. The Qubide may be ready by February 1994, at a price between £75 and £100. The closer they can get it to £75, says Ron, the happier they will be. The Qubide will drive one hard disk of up to 120 megabytes, which is plenty for nearly any application other than industrial cad or dtp, and more than enough for all the QL's memory-efficient applications. For more information contact **Qubbesoft** on 0376 890986.

MERZ'S NEW MULTITASKING CHESS

Jochen Merz Software have a brand new, multi-tasking chess program, **Black Knight**, for the Pointer Environment.

Black Knight runs with a 5000-move opening library, 10 levels of play from 5 seconds to 1 hour, and a chess clock. It can set up positions and load and save games.

Because the program will seize, in true battle fashion, almost as much memory as it can find to run the largest possible transposition tables, Black Knight benefits from hardware enhancements. However, it will run on a minimum of 640K memory, and will multitask with other programs (including itself) provided you tell it at the start how much memory you want to remain free.

The **Pointer Environment** (which is not supplied) must be installed before the game starts. Black Knight runs with the Gold Card, Atari STs with a QL emulator, and the QXL as well as other expansions with sufficient memory. TKII's default data directory can be used to load and save if it is present, but is not necessary.

Both the mouse and cursor keys can be used, and Black Knight will disallow illegal moves.

The exclusive distributors of Black Knight are **Jochen Merz Software, Im Stillen Winkel 12, D-47169 Duisberg, Germany. Tel. +49 203 501274.**

QL World has a copy of Black Knight for review. Please contact the Editor.

Welsh Board Tackle Spelling!

The Welsh Language Board has come up with a Welsh Language spellchecker, a gargantuan task when one Welsh verb can produce 100 forms. At present, the spellchecker can only detect routine typos - full error detection can only be achieved when a grammar-checker is incorporated as well, for reasons which we're sure a Welsh speaker would like to explain! Unfortunately, the first release is only for WordPerfect 5.1 on you-know-what, but maybe some enterprising supplier could build the vocabulary into one of our QL word processors. Qwyll, for instance?

For more information contact **Huw Owen, The Welsh Language Board, Longcross Court, 47 Newport Road, Cardiff CF2 1AD. Tel. 0222 488745.**



10 YEARS OF DIGITAL PRECISION TOO!

Digital Precision are celebrating their **10th Anniversary** in 1984. Digital, the longest-lasting QL software company, launched their range after many months of preparation with three programs: QL Super Sprite Generator, QL Super Monitor (with free 68000 Disassembler) and QL Super Backgammon. It's fair to say that these products nicely represented the interest of Freddie Vachha, Digital Precision's substantial genius loci, in practical utilities and serious game programs.

QL Sprite Generator and Super Backgammon are still in the catalogue, and Super Monitor has been replaced by QMon Machine Code Monitor. Over the years these three have been joined by no fewer than 80 other programs, from Lightning (for many their first experience of a fast QL) to Perfect Pointer Tools, the Perfection wordprocessor to the Conqueror PC emulator, Professional Astrologer to Eye-Q graphics, Turbo Basic Compiler to Micro Bridge.

Digital's first advertisement naturally went with a splash, a full-colour page with discounts for buyers of two or three programs, and an appeal for good original QL software. And there for the first time was the distinctive Digital Precision logo.

Extraordinary as it may seem now, in the early days Digital Precision proceeded gradually, advertising occasionally and building their list slowly. By the end of 1986, however, they had transferred from Manor Road to 222 The Avenue, where they still are today, and added Professional and Super Astrologer, Eye-Q, Supercharge, Super Media Manager, Superforth and several games. The rest, as they say, is history!

Scan Digital Precision's software menu on **pages 2 and 3.**

Open Channel

Open Channel is where you have the opportunity to voice your opinions in Sinclair QL World. Whether you want to ask for help with a technical problem, provide somebody with an answer, or just sound off about something which bothers you, write to: Open Channel, QL World, The Blue Barn, Tew Lane, Wootton, Woodstock OX7 1HA.

Non-Hexed

For people who don't want to try changing hex bytes to fix the Abacus AMEND bug in West Midlands Xchange, as described by Ron Stewart in QL World II.9, I am sending a listing that will fix it.

Howard Clase
St. Johns
Newfoundland
Canada

PS - Budgie?! That's a common or waterside Kingfisher! What did you get

in nature study?

Mostly disembodied bulls' eyes. The birdie is in fact Alcedo Atthis Barkerii, whose characteristics are a short beak and a garish habitat. It goes around in pairs, and lives on those little fish sometimes found swimming in monitors, and any bugs it can find. Every programmer should have one!

Downloadable Dumfounding

I would like to thank you and your team for a fine magazine. The articles are always meaningful to most of us ... have you ever read the average photo or radio magazine, or ...?

But I have some comments and hopes for the future. My set-up consists of a boxed QL with Gold Card and Minerva, with two 720K and two ED drives, external keyboard and the Mersey Mouse. The printer is a NEC Pinwriter P2200.

I have managed to get Wordperfect operational using the four drives (flp4_ as a pseudo hard disk), but I

still prefer Perfection SE and to use the QL as a QL!

Now for some time I have envied other people with their more modern printers having available the **extra fonts** built in. Being one of the thousands thrown onto the heap by the march of time and the latest technological innovations, I can't go out buying the latest gear or expensive programs. I've played around with some of the relevant progs from the Quanta library, and was impressed by the Borman..Fonts.

But it's obviously not as simple as that! having become a little disillusioned and not a little confused about the whole thing, not getting the results I wanted, I wondered if there's a chance that QL World might have a future series, including answering the following:

Just what is the nature of built-in fonts in a printer?

Do downloaded fonts to the printer look the same to the Ser port as inbuilt ones?

Can these be used in the same way (say, from Perfection) as the normal fonts, ie underlining and Bold?

Would it be feasible to use Olde English and the like in Perfection and get a printout in the same font?

Also, would it be possible to do a rehash (they must have appeared some time, somewhere) of listings to Search and Replace (strings on a disk file)? And disk comparison and file comparison type programs (right up Simon's street, perhaps). Again, many thanks for what we are getting. Seasons greetings.

Bryan Orgar
Ashford
Kent

Moving fonts from one source to another, especially one computer to another, can raise unpredictable bugs. Sometimes one has to refer to the makers of both for help. Apart from the famous (old) QL overheat, I have seen more systems (all kinds) crashed by font anomalies than any other cause! PD fonts give the most trouble. Fonts look superficially simple, but they are complex shapes and can be difficult for the operating system to handle. Oddly, simple fonts seem to

```

5 REMark A program to fix a bug in the West
10 REMark Midlands PD version of Xchange that
15 REMark causes a crash when the ABACUS "amend"
20 REMark function is used on a Gold Card.
25 REMark See QL World 1993 Vol2 #9 p 10
30 REMark H.J. Clase 1993.11.12
100 REMark ~~~~~~main
105 CLS: PRINT"Put your copy of Xchange into"
110 PRINT"flp1 and press any key when ready."
115 PAUSE (-1): OPEN#3,flp1_Xchange:
120 ad=41194: long_word$=Check_long_word$(ad)
125 IF long_word$="52390000"
130 PRINT "Bug found". BPUT#3,(ad),6,45,0,1
135 IF Check_long_word$(ad)="062D0001"
140 PRINT "Bug now fixed": END IF
145 ELSE IF long_word$="062D0001"
150 PRINT "Bug already fixed"
155 ELSE : PRINT "Not West Midlands' version"
160 END IF : END IF
165 CLOSE#3
200 REMark ~~~~~~
205 DEFine FuNction Check_long_word$(ad)
210 LOCal i,b$,a%: b$=""
215 FOR i=ad TO ad+3:BGET#3\i,a%: b$=b$&HEX$(a%,8)
220 RETurn b$: END DEFine
300 REMark ~~~~~~End

```


give as much trouble as fancy ones.

The office font enthusiast says: "A downloaded font will look just like an ordinary printer font. There will be a special control code to switch to the downloaded font, just as there is a control code to switch to the resident fonts.

On laser printers individual fonts are provided for bold, italic as well as standard. I don't know whether the bold and italic versions for the NEC P2200 are stored in the printer as separate fonts, or whether the printer simply approximates bold and italic itself. It is best to talk to the vendors (in this case Digital Precision) about fonts your wordprocessor can use."

I will make enquiries about the other matters.

Calling QL from Germany

On behalf of all our members we would like to

express our gratefulness to all Sinclair QL User Groups, Publications, Hardware Producers, Software Houses, Dealers and Distributors, for your continued support of the Sinclair QL and compatibles, with hardware add-ons, new software developments, new computers and especially all kinds of service.

The situation of the QL scene has dramatically changed in the last two years, and there is a certain ambiguity in this process.

On the one hand, the last two years have seen numerous new products, high quality software and several successors for the Sinclair QL are now being offered, and an excellent service on commercial and private basis, namely user groups, printed publications, and electronic networks.

On the other hand, the membership of user groups is slowly fading away, and despite all the modern technology available, many users are giving up the QL. It is our intention to bring

these two sides together by giving the highly active part of the QL scene a chance to demonstrate their developments and services to the more passive users, and we want to show pure users that the operating systems QDOS and SMS2/SMSQ have a future.

We also intend to bring together all sorts of people interested in the QL for informal exchange, to inspire each other and direct help. Finally, we want to provide commercial and private support organisations with the possibility of demonstrating their products or services.

This is to be realised at an International QL Meeting in Germany on February 19th at the University of Bielefeld, north-east Germany.

It must be clearly pointed out that this meeting has no commercial background, and that it is not profit-oriented. We are not charging an entrance fee, and fees for commercial exhibitors will be quite low. If you wish for more information, feel free to contact us for the full meeting information.

Franz Herrmann
Sinclair QL User Club eV
Talstrasse 21
D-53545 Ockenfels
Germany

POKE Modification

I've taken an interest in Dr. Teply's letter. I have not really been surprised, because a QLCF member previously told me that the trick doesn't work on his Gold Card (recent version) too.

My mind is that the POKE, POKE_W and POKE_L keywords work fine on my Gold Card (2.26, yellow pcb) in their original form, and doesn't work in the manner with some others GC roms.

I guess that Miracle have modified, for security reasons, the behaviour of these keywords, in order that it may not be possible to amend QDOS routines. If so,

they can't be blamed, because POKEing in this memory area is very risky, and a "bad" POKE can lead to the QL crashing.

Provided I've guessed right, the solution to this problem lies in the program:

```
100 base=ALCHP(98)
110 RESTORE
120 FOR offset=0 TO 96
STEP 2
130 READ mot$
140 POKE_W
base+offset,HEX(mot$)
150 END FOR offset
160 SBYTES
flp1_pokes_rsp,base,98
170 RECHP base
180 STOP
190 DATA
"3078","0110","43FA","003C","4
ED0","7800","6112","1881"
200 DATA
"4E75","610A","3881","4E75","6
104","2881","4E75","7801"
210 DATA
"3078","0118","4E90","6614","5
543","6610","2876","9800"
220 DATA
"2236","9804","200C","C084","
6602","4E75","584F","70F1"
230 DATA
"4E75","0003","FFC6","0550","4
F4B","4542","FFC6","0550"
240 DATA "4F4B","4557",
"FFC4","0550","4F4B","454C",
"0000","0000"
250 DATA "0000"
```

The program creates and saves a file which, once LRESPred, makes the keywords POKEB, POKEW and POKEL ready for use. This code is almost the same as the one which stands in the JM rom.

Could Dr. Teply use POKEB, POKEW and POKEL in lieu of POKE, POKE_W and POKE_L? I think this trick will also (for the same reason) allow him to modify Font 2 of the QL character set.

Please, somebody, let me know if the "cursor colours" trick now works, and correct mistake(s) if you then intend to publish this other trick.

Bruno Coativy
Rennes
France

Editor's notebook

Our lead article this month is a review of the newest upgrade to the long established QL flight simulator, Flightdeck, by Bernard Denchfield of Deltasoft. Ray Dawson not only assesses the program, but offers a guided flight for the newcomer, complete with rotation and navigation.

Bryan Davies is 'on holiday', partly due to pressure of work and partly due to the fact that his post didn't get through before Christmas. One of our contributors measured a time lag of nine days for a first class packet from Gloucester to Bedford. No doubt others have been experiencing the same delays. Our 'late bug' actually helped us get some stuff delivered this month.

Some late news about the workshop meeting in Edinburgh on February 12th dropped onto our doormat this morning (we still haven't seen it in Quanta), so the best course for anyone in the locality who wants details in a hurry is to contact Dr. Alan Perberton at Napier University, Craig Lockhart, Collington Road, Edinburgh. Tel. 031 660 1826 (evg) 031 650 88920 (day).

Last but not least - Happy New Year to all our subscribers, readers, and helpers. Here's to 1994 and next Hogmanay.

FLIGHTDECK - 1.03

**Ray Dawson
takes off into the
wild blue of
Manchester.**

INFORMATION

Program: Flightdeck V1.03
Publisher: Deltasoft via DJC,
41 Bro Emrys, Tal-Y-Bont,
Bangor, Gwynedd LL57 3YT.
Tel. 0248 354023.

Price: £15 cartridge or 3.5 in
disk. Will run on unexpanded
QLs.

Written mainly in machine
code, Flightdeck allows you
to "fly" a twin-engined jet air-
liner, and provides 3D views of
the "world" outside and a
database of 25 UK airports
and over 200 navigation
beacons. The plane has twin
VHF omnidirectional range
and distance measuring
equipment (VOR/DME), auto-
matic direction finding (ADF),
and instrument landing sys-
tem (ILS).

If you want to fly a light air-
craft under the bridges of the
River Thames, this program
is not for you, but if you want

to fly by instruments to major
UK airports, this is your pro-
gram. If you have watched
the Boeing 737 flight simula-
tor on The Krypton Factor on
tv, then this is perhaps the
easiest way to visualise
Flightdeck. In the tv pro-
gramme, contestants must
land the aircraft using only
instruments. Flightdeck's
graphics are not in the same
class as the 73 simulator,
but a brave effort has been
made by Bernard Denchfield
to provide a meaningful view
from the window, within the
constraints of cost and
graphics.

This view can be widened
by 30 degrees in each direc-
tion by pressing the appro-
priate keys - just as if you
were turning your head from
side to side in a real cockpit.
At startup, the pilot is provid-
ed with a screen to set up

amount of fuel, aircraft take-
off weight, sound effects
(on/off), and performance
limits (on/off). This latter
option provides for future
expansion, but I cannot think
what form this would take!
Pre-flight checks are last.

Octas and Cloud

The 2- and 8-octa cloud-
bases refer to the cloud
cover in "one eighth of the
sky" increments. One octa is
one-eighth cloud cover, 2
octas is two-eighths, and so
on. 8 octas means complete
cloud cover. Setting the 2
octas level at 6000 ft, say,
means that above 6000 ft
one would expect to be in
cloud 25% of the time, the
cloud gradually increasing
until 100% cloud cover
would be encountered at,

Runway when looking in a
westerly direction - the 24 is
the magnetic compass bear-
ing of the runway, cut to two
figures and rounded to the
nearest 10 degrees. The
actual heading of this run-
way is 238 degrees. The re-
ciprocal bearing is 058
degrees, ie "Manchester 06",
- the same runway from the
opposite direction! When a
landing is made on the ILS,
the instrument displays the
magnetic bearing 238 at the
top as the selected Radial
Beam, and its reciprocal 058
at the bottom to verify this.

The runway's magnetic
heading is essential for a
correct and successful
approach and landing.

I found the starting altitude
particularly useful, especially
to avoid doing "take-offs"
when practising landing, and
you will need a lot of prac-

FLIGHTDECK V1.03	
WIND SPEED.....	0 KNOTS
WIND DIRECTION.....	332
2 OKTAS CLOUDBASE.....	6000 FEET
8 OKTAS CLOUDBASE.....	8000 FEET
LOCATION.....	MANCHESTER 24
ALTITUDE.....	0 FEET
HEADING.....	238
PASSENGERS.....	100
FUEL.....	11000 KG
AIRCRAFT TAKE OFF WEIGHT.....	47000 KG
SOUND EFFECTS.....	ON
PERFORMANCE LIMITS.....	ON
PRE FLIGHT CHECKS.....	IN PROGRESS

Copyright (c) B. H. Denchfield 18 December 1988

the parameters for the flight.
- see **Figure one** for the
menu - using the up/down
cursor keys and the left/right
cursor keys. The flight is initi-
ated by the space bar.

The selectable parameters
are windspeed, wind direc-
tion, 2 octas and 8 octas
cloudbase (more of this
later), location, altitude, head-
ing, number of passenger,

say, 8000 ft, if that value had
been chosen as the 8 octas
cloudbase setting.

The start location must be
selected from the database
by cycling through with the
left-right cursor keys. Some
25 UK airports are included.
Different runways may also
be selected. For instance,
"Manchester 24" is the main
runway at Manchester

tice before you can execute
a perfect landing. This option
puts you at a pre-deter-
mined height above the air-
field and on a pre-selected
heading. Then you fly off for
about ten miles, do a 180-
degree turn and, having set
the correct instrument
Landing System frequency,
NAV1, make an approach
and landing, using the ILS

Instrument Landing System.

The sound effects can get a bit wearing, but I found them useful initially, as the engine noise saves looking continually at the throttle indicators to establish the thrust from each engine and adds a sense of reality. The squeal as the tyres touch down also adds reality on landing. The alternative - if you've made a hash of the landing - is a pair of black "curtains" closing across the window, with by a cryptic message such as "Crash - Undercarriage Not Down" or "Crash - Nose Wheel Collapse"! Fortunately, on a simulator one can always press the "Angel switch" and put the aircraft safely at the beginning of the simulation.

Manchester 24

A glance at **Figure two** will give you some idea of the Instrument Panel and the view of "Manchester 24" runway before takeoff. The panel is very well laid out, with 8 main dials. Engine, flap and landing gear indicators are in a "box" at the upper right of the panel. The navigation aid frequencies, NAV1 and NAV2 plus the DME1 and DME2 readings of the distance measuring equipment, along with ADF1 automatic direction finding beacon frequency, and the Fuel Indicator, are in another "box" at the bottom right. It is the VHF omnidirectional range, VOR1 and VOR2 frequencies, which are displayed as NAV1 and NAV2 in this box. DME1 and DME2 indicate the distances from those beacons. This enables you to get an accurate fix of your position, providing the aircraft is within range of the right beacons. If it is out of range, the VOR instruments display a red light to show that they are inoperative. The top left dial is the air speed indicator (ASI), calibrated in knots and with a digital readout at its centre. This measures the speed through the air (as opposed to speed over the ground) -

there is a difference due to wind.

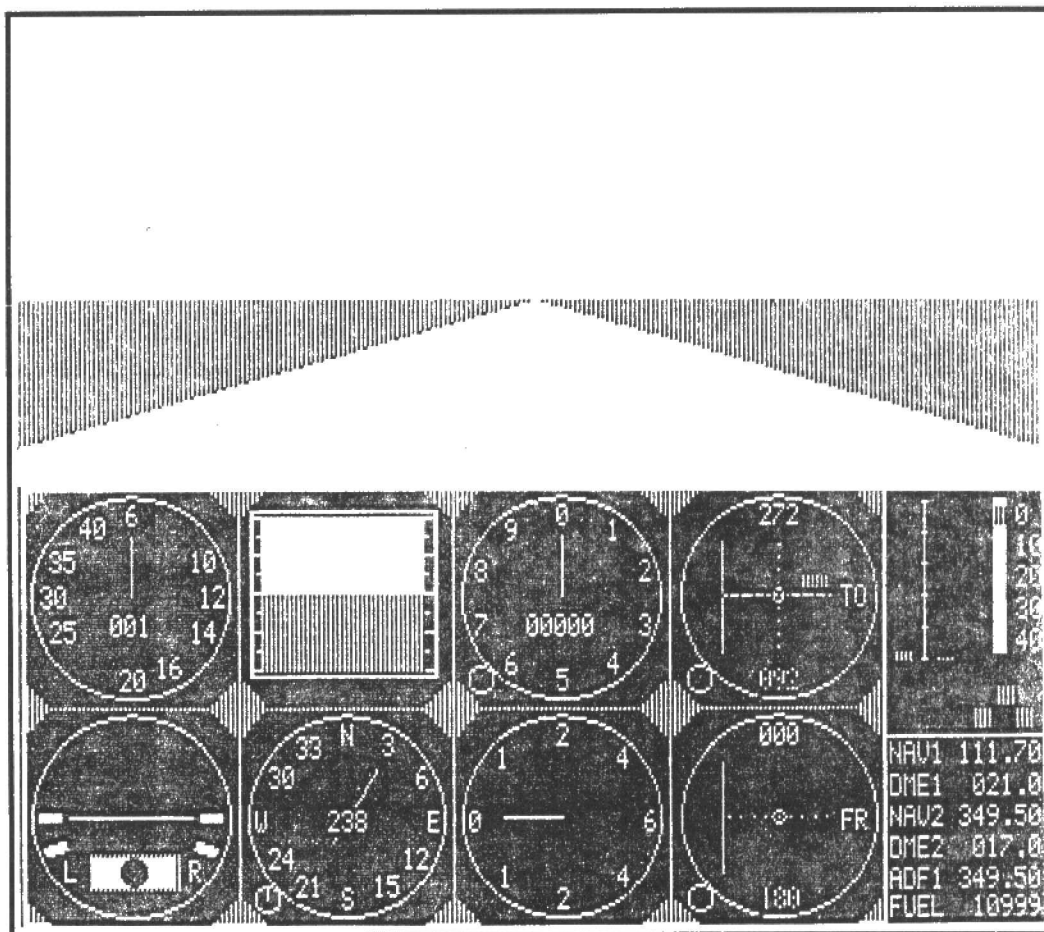
The artificial horizon is a square window with calibrated scales down each side. A shaded area in the square gives the aircraft's altitude. The larger scale markings represent 10 degrees of inclination, the smaller divisions represent 5 degrees within a maximum of plus and minus 20 degrees. If you can imagine looking straight ahead through a square window in the cockpit: in level flight, the horizon would appear to be across the middle of the window and horizontal. If you are fly-

Next comes the Altimeter, calibrated in feet with a digital readout at its centre. It gives the aircraft's height above ground. (All airfields are assumed to be at sea level. This eases the burden of adjusting the altimeter for each airfield's height above sea level.)

On the extreme right is the VOR1 indicator (VHF omnidirectional range number 1) which also doubles as the instrument landing system (ILS) indicator, and this will be covered later.

The bottom left hand instrument is the turn coordinator. This indicates the

handled by the artificial horizon. Incorporated in this instrument is the balance indicator, which is used to determine if the aircraft is "skidding" or "slipping". This instrument contains a ball in a tube, which is displaced sideways by a sideways force, as may occur by applying rudder without an appropriate degree of bank, causing the aircraft to "yaw" or go into a flat turn, a most uncomfortable feeling. This rarely happens with this aircraft, as all turns are co-ordinated using aileron only, the correct amount of rudder being applied automatically.



ing "right way up", then the ground appears below the horizon and the sky above. The ground is shaded green and the sky white. If the aircraft is "nose up", the horizon moves upwards towards the top of the window. If the aircraft banks, the artificial horizon tilts, the degree of bank being shown on the scales at either side.

rate of turn by means of a tilting horizontal line, similar to the artificial horizon. This line represents the aircraft's wings. The instrument has calibrated marks to indicate a Rate 1 turn, that is to say, a turn of 3 degrees per second change in the aircraft's heading. However, this instrument does not indicate the angle of bank, which is

The effect of skidding or slipping can only be experienced by using the rudder alone, causing it to skid, or by applying too much bank for the aircraft's speed to support, causing it to slip and possibly stall. There is an audible stall warning device. If this is ignored, then a "stick pusher" automatically prevents a stall by push-

ing the stick forward to put the nose down. The air speed then increases and prevents the stall. You can still crash, though!

Next comes the automatic direction-finding (ADF) and compass. The compass requires little further comment, but the ADF beacons do. Each beacon transmits a signal on its specified frequency, at an equal strength to all points of the compass. The ADF needle in the aircraft always points to the beacon. It points vertically upwards when flying directly towards a selected non-directional beacon. If the

downwards, as the beacon is now behind you.

Climb and Descent

Next is the vertical speed indicator (VSI). This indicates the rate of climb or descent, and is calibrated in thousands of feet per minute. When the needle is above the horizontal, the aircraft is climbing, and when below, it is descending, the rate of climb or dive being indicated on the marked space.

The bottom right hand instrument is the VOR2 (VHF Omnidirectional Range

radial and fly towards, or away from, the beacon. The VOR/ILS beacon for each airfield is on the extended centre line of the runway so that flying along this radial, towards the airfield ensures the aircraft is on the correct heading. The ILS beacon also has a vertical fan shaped set of beams, to enable the aircraft's height, relative to its distance from the airfield, to be monitored. One of this set of beams enables a 3-degree "glide path" to be defined to assist a perfect approach and landing. Deltasoft are to be congratulated for a smart bit

airports, but the same rules apply when flying a short hop, or a much longer flight from Boumemouth to Aberdeen. It is much more realistic if you have got an air map of the terrain. If not, a road map will do.

Take a Map

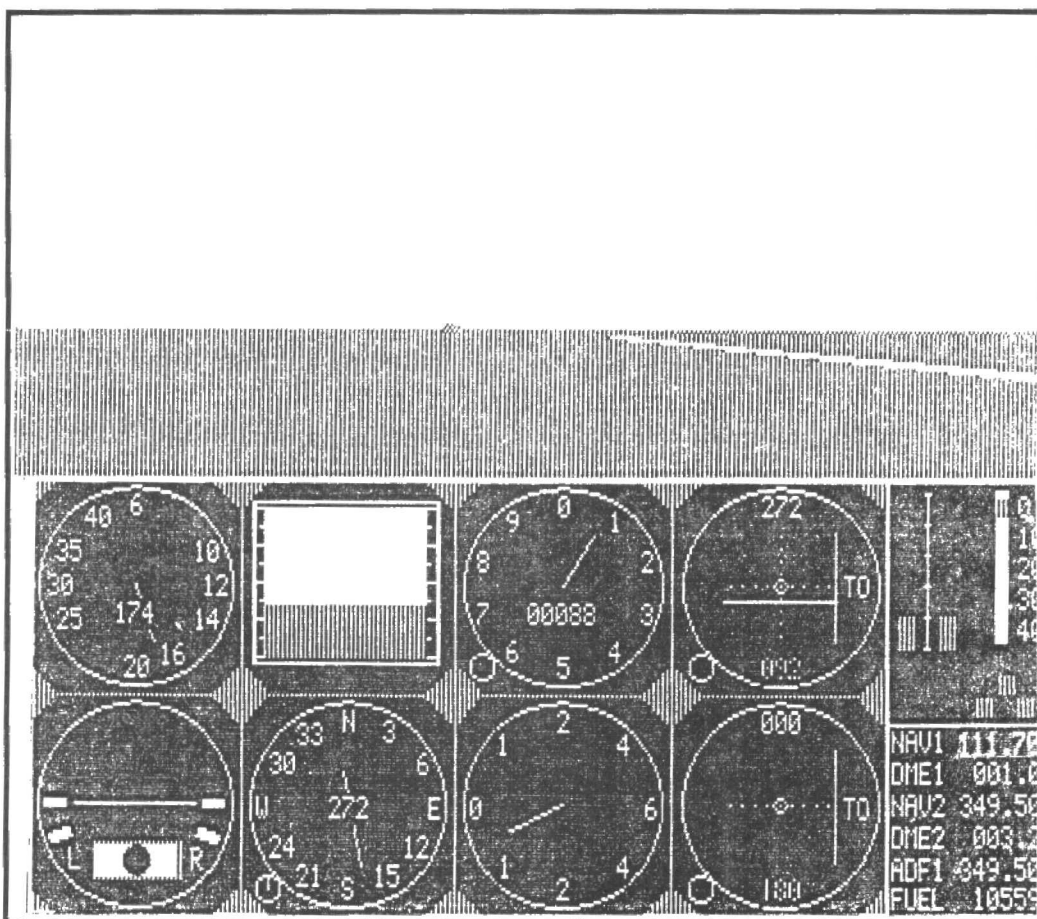
Draw a pencil line on your map from the start point to your destination, and measure its true bearing with a protractor. Convert this to a magnetic bearing by adding the magnetic variation, currently about 6 degrees west, and so arrive at the compass "course to steer". Look along this line and establish which ADF or VOR beacons you need. If you're clever enough to be able to navigate in windy conditions, set up a wind speed and direction from the opening menu. Make the corrections to your heading if you do this.

We will assume no wind and a clear sky and set the 2 octas cloudbase at 6000 ft and 8 octas at 8000ft. We will fly at 4000 ft, so the trip will be cloudless. Set the location to "Manchester 24", altitude to 0 ft, and the heading will automatically set to 238 degrees. Leave the passengers, fuel and takeoff weight as default, and move the cursor to preflight checks.

Look up the frequency for Liverpool 27 ILS beacon - Ident ILQ - and get ready to set it on the ILS instrument, VOR1. Having pressed Space to start the flight and entered these frequencies as NAV1 and NAV2, check the Idents to ensure you are correct. Setting ADF1 to the same frequency as VOR2 may help navigation.

Simulators Only!

If you do any real flying, on no account must you use any of the VOR, ILS or ADF frequencies listed, or anything else in this program, while actually in the air. These frequencies are for



needle moves to the right (clockwise), then the aircraft must be turned to the right to bring it back onto the correct heading, and conversely for the left. The aircraft's magnetic compass heading is displayed at the centre of this instrument. When an ADF beacon is overflown, the needle swings through 180 degrees and points directly

Number 2), and is similar to VOR1, except that it does not have the additional ILS facility. These beacons send "radial" signals just like the spokes of a wheel. Each beam is separated by 1 degree so there are 360 of them in all. By adjustment of the VOR, you can identify which radial the aircraft is crossing and turn on to that

of programming here.

Perhaps the easiest way of learning about this is to do it. Climb aboard! Strap yourself into the right hand seat. I will be in the left hand seat as the Captain, and we will go from Manchester to Liverpool. You have already guessed that I come from the North by my accent, and I have selected two northern

simulator use only - you have been warned. Do a last check, take a look around to see the sky and runway are clear, select 10 degrees takeoff flap, open the throttles together slowly and evenly to full thrust.

As the aircraft starts to move, keep straight and level, using the rudder, if necessary, watching the airspeed indicator. With 100 passengers on board and 11000 kg of fuel, the "rotate" speed (take-off speed) will be about 150 knots. As this approaches, pull backwards steadily on the control column and the aircraft will gently become "unstuck", and climb. Hold this position until the artificial horizon indicates about 15 degrees of climb and then move the stick forward to maintain this attitude. Check the climb-and-dive instrument for a positive rate of climb, and raise the landing gear, checking that the gear warning lights go red to confirm this. As the speed passes

240 knots, raise the flaps.

While climbing, ease back the throttles to about 70% thrust and turn the aircraft to its correct heading as per your flight plan, in conjunction with the navigation instruments. In a short time, VOR1 and VOR2 will indicate that they are operational (the red warning lights will go off) and the DME (distance measuring equipment) on VORs 1 and 2 will show the distance to your destination. These are not the same because one is the ILS beacon and the other is the VOR, a few miles apart - the ILS beacon is lined up on the runway and the other some distance away in the fields. Level off at about 4000 ft.

Try to check which "radial" you are crossing via the VOR instruments. It is important to get the final turn on the correct heading for your approach. In a big aircraft the controls will take effect slowly due to inertia. About 10 miles from Liverpool start

descent by slightly reducing thrust and lowering the nose gently. Balance a reasonable rate of descent against throttle settings. When you are below about 235 knots, lower flaps and undercarriage, checking the warning lights. Raise the nose slightly, as increased draft from the flaps and undercarriage will enable an even descent with the nose about 3 degrees up. Adjust power and pitch attitude to keep the cross wires on the ILS aligned on the graticule. If the vertical wire is to the left of the centre line, you must do a gentle left turn to correct it. Do not chase the instrument backwards and forwards - make small corrections and wait for them to take effect. You should now see the runway, but watch your instruments. The ILS will assist you with correct height and heading, but you must control the airspeed. The landing speed is about 150 knots, so take this into account as you approach

the runway. Fine adjustments to the controls will bring the aircraft safely to the threshold of Runway 27. "Flair out" by raising the nose slightly, then close the throttles and let the plane sink gently onto its main wheels. When you hear the wheels touch down, ease the stick forward to bring the nose wheel down and gently apply the brakes.

Well! Height not too bad, but the ILS needle says "fly right", and you ignored it. I could go on but this is your first attempt. Many enjoyable hours are yours, but be warned - it's addictive.

The more technically minded can extend the database of airfields and beacons. A full description is supplied. If I ever manage to execute two consecutive perfect landings, I may be tempted to try a few changes but, up to now, I have been fully occupied. I cannot resist the temptation, so here goes ... Happy landings, folks! Have fun.

Telephone: 0753-886866 (EEC) W.N. Richardson & Co.
Fax: 0753-887148

SINCLAIR QL PRICE LIST DEC '93

All Prices Include 17.5% VAT

ASK FOR DETAILS OF ECONOMY RANGE OF F.D.D. UP TO 4MB ANT. H.D.D. UP TO 44MB.

IS COMPLETE: QL Computer, PSU, T.V. Lead, PSION V233 Software (Word Processor, Spreadsheet, Database & Business Graphics), Handbook for QL Programs, & Superbasic. JS £120

(A FEW LEFT) Free one year membership to QUANTA, the independent QL User Group with over 400 free library programs, Newsletters & HELP. 6 MONTHS WARRANTY. WITH JM ROM £100.00

PART EXCHANGE £10 OFF ABOVE. Send in old QL (Unit only, any condition). JS ROM £70. JM £70.00

BACKUP DISK QL & PSU only. JS £80. (Part exchange allowance £15 if required)

Accessories * NOTE: EXTERNAL (SER2) 3 BUTTON MOUSE AND SOFTWARE WITH EXTRA FUNCTIONS. NOW HERMES COMPATIBLE *

PC KEYBOARD INTERFACE, for INTERNAL FITTING, with lead to 102 KEY KEYBOARD £75.00

PC KEYBOARD, UK version, 102 key (AT) £25.00

PC KEYBOARD INTERFACE AND KEYBOARD → REDUCED PRICES → £95.00

CASE AND LEAD for EXTERNAL FITTING of Keyboard Interface for simpler assembly £14.00

JOYSTICK with QL lead (no interface required) £10.00

* QL MOUSE, 3 Buttons, Software controlled, externally connected, simply fits in SFR2 £45.00

TANDATA MODEMS 1Mb, 2Mb and 4Mb AGAIN IN STOCK

ECONOMY MULTI-COMPUTER FLOPPY DISK DRIVE

ECONOMY DRIVE PRICES INCLUDING VAT @ 17½%

CAPACITY:	1Mb 525D/2	2Mb 525D/2	4Mb 525D/2	Duplex2 1Mb-2Mb	Duplex4 1Mb-4Mb
SINGLE DRIVE	£69	£85	£120		
TWIN DRIVES	£99	£150	£199	£135	£175

TELEPHONE RE: UPDATING EXISTING DRIVES. ALSO DETAILS OF HARD DISK DRIVES

Monitors PHILIPS 14" HIGH RES COLOUR EGA .31 DOT PITCH, FULL 85 COL WITH AMBER OR GREEN TEXT FEATURE, IDEAL FOR WORD PROCESSING, RECONDITIONED, 90 DAY WARRANTY. £149

Microdrive Cartridges and Spares "MONO OPTION" WITH 9" GREEN SCREEN

4 New Cartridges in a wallet		£10.00	2 GREEN SCREEN		
Plastic Storage filing box including 20 new cartridges			8 prog. cards for reformatting in 2 wallets		£15.00
QL Psion Software V2.15 Includes Quill, Abacus, Archive, and Easel IN WALLSET					£45.00
QL Psion Software Separate programs					£10.00
QL power Supply Unit		£10.00	QL Printer 1/8"		£27
TV or A-network leads		£3.00	QL Top & Bottom Case		£5.00
IC's ZX 801		£9.00	ZK 802		£3.00
			MC 1377		£3.00

QL SERVICE MANUAL & CIRCUITS £25.00 S.A.E. FOR FURTHER DETAILS

Payment terms: CWO, Access, VISA as terms Cheques - allow 10 days
Product subject to availability. E&OE
TEL: 0753 888866

Delivery: Carriage - £9 Postage - £5
MOBILE: 0850 597650 FAX: 0753 887149

TF SERVICES

MINERVA

The ULTIMATE operating system upgrade

MKII MINERVA with buffers for 256 bytes ram, CRASHPROOF lock & Philips I-C bus for interfacing. £29.95 (includes 100% backed ram, Quick start-up)

Other features common to MKI/MKII

DESH GOLD operating system autoboot on reset. Multiple Basic fast scheduler graphics (10% of 1/2min) string handling WHEN ERROR/2nd screen. TRACQI (on-keyboard) drivers "warm" fast reset. Auto reset after power failure V1.97 now with built in Multitask and split OUTPUT hand rates with Hermes.

1st upgrade, free. Otherwise £15 for manual if required (send SAE, Minerva & NEW disk 3 inches). MS1 to MKII upgrade £40

MKI... £35 MKII(RTC)... £60

GOLD CARD (x2.24) COMPATIBLE

HERMES

A replacement QL co-processor for the QLs awful IPC 8049

Do you get keyboard bounce?
Do you find fast serial input unreliable?
Do you want to connect a modem at 19200bps

If you can say one YES, then you need HERMES

19200bps RELIABLE serial input - NO QCONNECT!
Independent input hand rates - use serial mouse & print
Stops keyboard bounce (forwarded report chrs)
Improves "noisy" and "random" sound
Provides extra input/output lines
Key click

Fitting is simple. Remove the QL top (8 screws) & replace the chip marked 8049 or 8749 next to mdy 1

£25 including manual/software

QL SPARES

Family QL board (no glue-in chips) £9
Keyboard membrane £9 Circuit diagrams £2
68008 cpu £8 1377 PA1 £3
JM ROM £10 Power supply £12
80211A £10 80211A £10
8049 IPC £8 MDV (1A) £12

Other components (sockets etc) please phone

QL REPAIRS

Fixed price for unmodified QLs, excluding microdrives. QLs tested with Therm-EMI fix and ROM software

£27 including 6 month guarantee

Prices include post & packing (UK only). Payment by Mastercard/Visa/Access/Eurocard/cheque/postal order/PO. Giro transfer (58 267 3909) MAIL ORDER ONLY - no callers without ringing first. Ring for overseas prices

Holly Corner, Priory Road, ASCOT, Berks, SL5 8RL
Tel: 0344-890986 Fax & BBS: 0344-890987



Keyboard-90 Interface

Alex Munden offers a user report on his "Keyboard for the '90s".

INFORMATION

Manufacturer/supplier: Jurgen Falkenberg Computer Technik, Thanweg 361D-7539 Ersingen, Germany. Tel/fax: 010 49 7231 81058

UK Supplier: W N Richardson & Co (EEC), 8-21 Misbourne House, Chiltern Hill, Chalfont St Peter, SL9 9UE. Tel: 0753 888866. Fax: 0753 887149.

UK Price: Interface only £75.00 + p&p, keyboard only £30.00 + p&p, Interface and keyboard £95.00 + p&p, External fitting kit £14.00 + p&p.

Throughout QL history, a number of replacement keyboard systems have appeared but, for various reasons, have fallen by the wayside. Now Jurgen Falkenberg's Keyboard-90 interface allows any AT- or XT-style PC keyboard to be connected to the QL.

No nightmare

At first sight, it looks like an electronics novice's nightmare, comprising a 120x55 mm fibre-glass printed circuit board, ten ICs, a handful of resistors, diodes and capacitors, a crystal, a DIP switch and an empty 40-pin IC socket. Also attached to the board is a length of cable and an in-line DIN socket. The amount of work required to convince the QL that it is still connected to its native keyboard can be seen in the form of a 32K eeprom labelled QL-T90, which contains the firmware, and an Intel 80389 co-processor. The remainder of the chips are electronic latches and switches.

Internal installation of the interface may look like a major obstacle to anyone who has never seen the inside of their favourite black box, but things are not as complicated as they seem. What is mainly required is

confidence and patience. The interface board displaces the Intel 8049 chip, the QL's door to the outside world. This explains the empty socket on the interface board. Removal is best done using an IC extraction tool or a flat bladed screwdriver. IC pins are quite soft, and can snap off very easily if bent or displaced.

You are then left with the problem of what to do with the DIN socket and its trailing lead. In my case, I found that, with a bit of persuasion, it could be eased into the slot in the casing alongside the tv modulator socket.

Then comes the moment to plug your PC-type keyboard into the DIN socket and switch on. If everything is working correctly, the normal start-up prompts should appear on screen. If not, press the reset button and try again. Your QL should respond to the new keyboard as though nothing had happened. If the QL still sulks, something is obviously wrong. Before you go any further, check that the pins on the interface board and the displaced 8049 chip are correctly aligned in their sockets. An error here is the most likely cause of failure.

Complete

Now for the good news. For anyone who has just read this with a sinking heart, a case and lead for external fitting of the interface are currently available from the UK supplier.

Anyone who has ever encountered a PC keyboard may be wondering how it relates to a QL. By now you should be familiar with the standard Qwerty pattern, but several key positions differ from the QL layout. If you purchase your keyboard with the interface, this should not be a problem, as new keytops will

already have been fitted. If not, it's worth pointing out that the "@", "#", and copyright symbols are where you would expect to find them, but the pound sign sometimes nestles alongside the Enter key. The cursor keys are grouped together to the left of the main keyboard, and there are two extra Control and Alt keys, on opposite sides of the keyboard. They just duplicate each other (like the pair of Shift keys.)

All the extras!

Twelve function keys form a row along the top of the keyboard. F1 to F5 behave as you would expect, and F6 to F10 are coded as Shift-F1 to Shift-F5. F11 and F12 are the equivalents of Ctrl-F1 and Ctrl-F2 and could easily be detected by software. Also in this row are the ESCape key, and three keys marked respectively Print Screen/SYSRQ, which generates Ctrl-C; Scroll Lock, which returns Ctrl-F5 and Pause/Break, which is the equivalent of the QL's Ctrl-Space. To the right of the number keys is a back-arrow key, which duplicates Ctrl-left cursor as a delete-left key.

To the right of the main keyboard is a group of six keys marked INS, Home, PG Up, Del, End and PG Down, or, if you are lucky, more coherent equivalents. INS is the equivalent of Alt-Enter, Del replaces Ctrl-right cursor as the delete-right combination and Home, End, PG Up and PG Down are the respective equivalents of Alt-left cursor, right cursor, up cursor and down cursor. These have little bearing on Psion software but many newer programs use these combinations for screen navigation.

Finally, to the far right of the keyboard, is a 17-key numeric keypad. When Num Lock is

pressed, an indicator led lights up, and the pad accepts numerical input from 0 to 9, plus the decimal point, the mathematical symbols and Enter. With Num Lock switched off, these keys duplicate the cursor and Alt-cursor keys, together with INS and Del.

Advantages

So, what's so special about attaching a PC keyboard to your QL? To begin with, there's the quantum leap from 65 to 102 keys. The keys themselves are properly angled for more comfortable input. Anyone who has spent hours slaving over a QL keyboard will know the strain it can place on an accomplished touch-typist, let alone the average "hunt and peck" operator. Keystrokes are more positive and give a greater degree of feedback, especially for non-typists who spend most of their time looking for keys, rather than confirming that key presses have been accepted on screen. The Caps Lock indicator light is a great improvement, particularly for those of us who still use Psion software. The single key replacements for many multiple keystrokes are a real boon and the separate numeric keypad really speeds up repetitive input, even for the 10% of the population who are "dextrally challenged". (Left handed, for those who did not realise that PC no longer means personal computer!)

All in all, adding a "proper" keyboard to a QL gives it a new lease of life. It's a bit like getting to know the QL all over again, now that you have all the benefits of Qdos and real multi-tasking, coupled with the advantage of a keyboard that does what you ask it.

5th italian show

Despite torrential floods and a national railway strike, QL enthusiasts from Italy and the UK flocked to Reggio Emilia for the 5th Italian QL Show this autumn.

The traders included Tony Firshman, Bill 'EEC'

Richardson, Ergon Development, Miracle Systems, SPEM, the QItaly user-group and Qubbesoft. The venue was a large room on the top floor of Reggio's modern Civic Centre, with that all-important en suite pizzeria. Admission was free to both traders and customers.

Roberto Orlandi and Eros Forenzi of QItaly were among the first to arrive, with an impressive A4 magazine

QL Mondo, copies of their disk newsletter, QPac upgrades, QL books in Italian and many items of hardware and software for sale.

These two are Qdos fanatics, with 'QL' stickers on the backs of their cars which confuse lesser Italians, for whom 'QL' is an abbreviation for a fraction of a ton. When Computer Shopper had a QL column Eros used to drive 50 kilometres to Switzerland each month for his 88p copy!

Since the demise of Sandy, SPEM have been

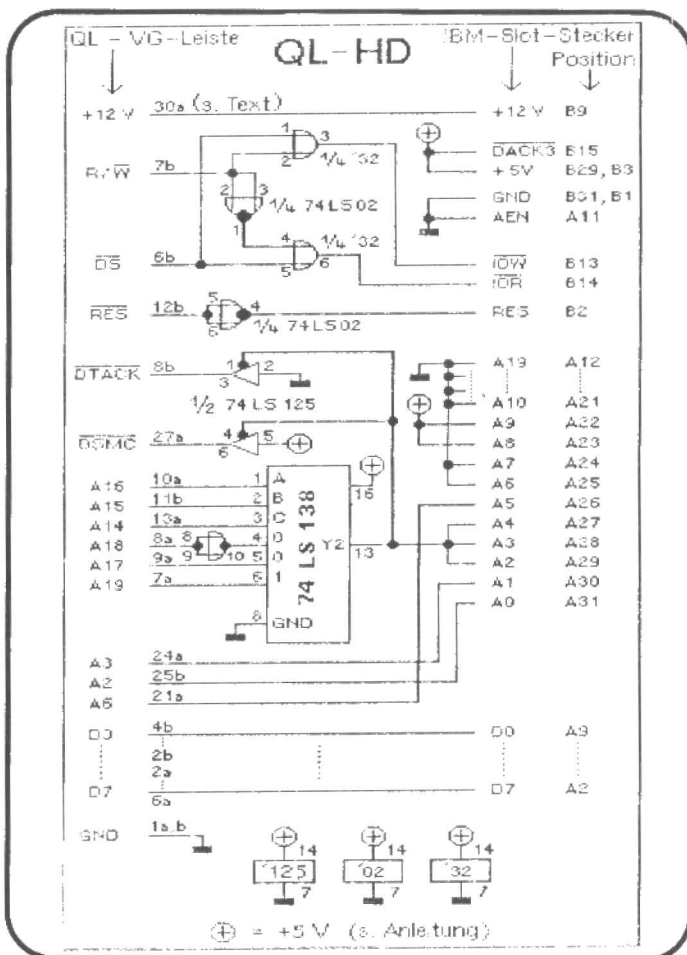
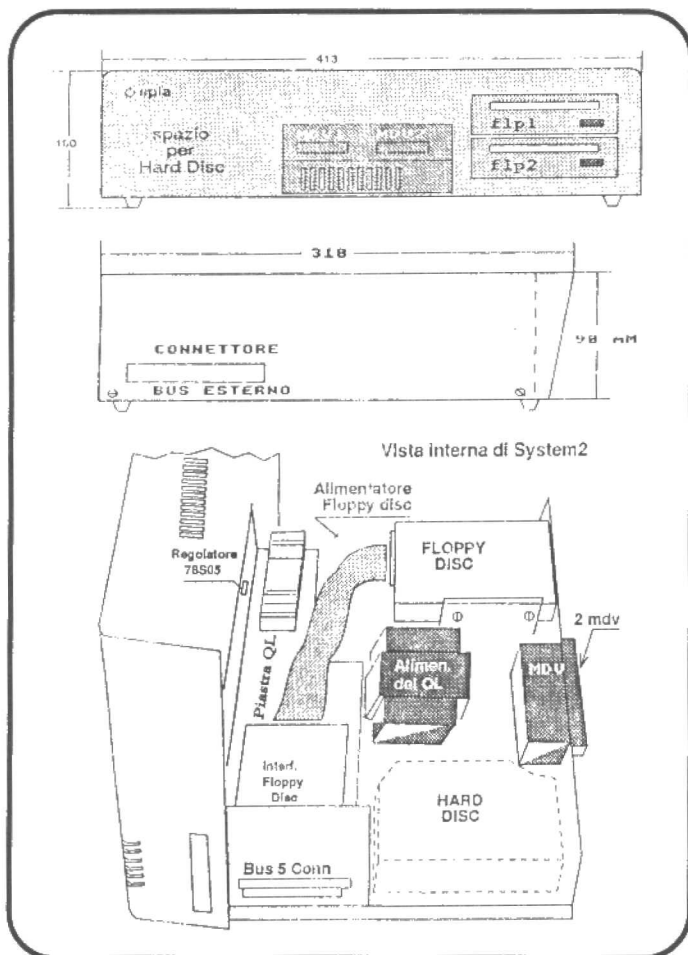
kings of the Italian QL hardware scene. They were showing off their QL expansion keyboard, which has gained a new lease of life since the arrival of Hermes, and their impressive System2 QL expansion chassis.

This is a white metal box with a flip-top lid, with fittings for the QL circuit board, hard disk, two floppies, two microdrives, power supply and all your interfaces. With the exception of the microdrive extension port, all the Sinclair connectors remain available around the edge of

the chassis.

System2 found users on several stands at the show, alongside custom variants, QXL-powered PCs, and original QLs. Even one shy Amiga 500 made an appearance. If you have no qualms about sawing your QL in half, to extract the microdrive mountings, the System2 is almost certainly the QL box for you. SPEM also sell a range of expansion boards and EPROM cartridges.

Ron Dunnett, Tony Firshman and Stuart Honeyball had planned to



travel by train and their customary folding bicycles. Alas Italian railway workers took Sunday off in protest at impending privatisation. The contingent arrived, sleepless but intact, after an eventful 17-hour drive from Paris, dodging storms, Alps and flooded motorways.

I arrived by plane and train a couple of days early, the guest of Ergon boss Davide Santachiara, and spent an entertaining weekend trying to coax secrets of the **ZM/HT compiler** from its enigmatic co-author Marco Ternelli. Ergon's best-seller at the show was their new Disk Utilities, capable of repairing Gold Card HD and ED disks, and formatting HD to over 1.52 megabytes. Gold Card and Spectrum fans upgraded to the new ZM/128 emulator.

Stuart Honeyball of Miracle Systems gave a talk on QXL and his plans for the future. The next release of QXL will include network and serial

device support, with floppy-disk formatting not far away. The new SuperBasic interpreter has been delayed, because Laurence Reeves has broken his arm.

Work on the 'low-cost' QL SCSI interface is well underway, but Stuart told the show that the QL graphics board is on the back burner till the QXL software is more complete. Meanwhile QXL users can expect regular software updates.

Tony Firshman demonstrated the first production peripherals for his **Minerva I2C interface**. These black boxes are labelled in red and festooned with D-type sockets. They are controlled by the same Philips circuits as Minerva's Mark 2 clock chip, and you can connect up to eight interfaces of each type to one QL. Large collections will need an external power-supply.

The eight-bit Analogue interface has eight inputs and two outputs. The sim-

pler digital interface has sixteen five volt outputs which can be turned on or off from SuperBasic, or re-programmed as digital inputs. High power versions for AC and DC control are coming soon.

Bill Richardson, formerly of EEC Ltd, brought hardware and brochures from the UK and Germany, most notably including Jurgen Falkenburg's **QL hard disk adapter**, which uses a PC-XT interface and drives. This really needs a box, but should still find users as it's the only plug-and-go hard disk system currently on the QL market.

Impecunious enthusiasts may favour Dirk Steinkopf's **DIY approach**, which uses just four TTL chips, worth about a pound, to adapt the QL expansion connector signals to a simulated PC/XT slot. A full 68000-code device driver is on disk, with a circuit diagram, utility programs in C and

documentation in German and English.

Dirk's kit has been tested with OMTI and Western Digital hard disk controllers, specified in the documentation, and suits both MFM and RLL drives. It is said to co-exist happily with Gold and Trump Cards and supports QJump level two devices, with subdirectories. My copy came from Ergon, and is freely distributable; you can get it from bulletin boards, user groups and PD libraries, such as **Qubbesoft's PD disk SPECIAL 22**.

Participants found it easy to communicate in a mixture of English, Italian, Latin and French, and it is a pity there were not more people from Northern Europe at the show. Perhaps storms and railways defeated them. On this showing, the QL is keenly and imaginatively supported in Italy; the next meeting should be just as rewarding, and hopefully a little sunnier.

Now available on a QL near you...

GT-Prolog/QL

**Edinburgh Syntax
Incremental Compiler
Automatic Garbage Collectors
Tail-Recursion Optimisation
First Argument Indexing/Hashing
32bit integers / 48bit reals
255 byte atoms / 32kb strings**

Interactive Workbench (needs 512k)

*** Windows, Menus, Dialogues**

*** Editor**

*** Debugger**

User Guide, Reference Manual

Full QDOS compatibility

Configurable Memory up to 16Mb

Turbo Performance - Naive Reverse exceeds 5k LIPS (GC)

Price including postage/packing £89.95

For more information contact us at
**Rosebank, Stream Road,
Upton, Oxon, OX11 9JG
Tel/Fax: 0235-851818**



**Grange
Technology
Limited**

DIY TOOLKIT

Simon Goodwin teaches the DIY Mouse to emulate keyboard entry for older programs.

The small MOUSE_PAINT listing is a simple mouse-driven painting package in SuperBasic, which uses the DIY Mouse extensions. It lets you move a cursor anywhere on the screen, leaving a trail of dots in one of the eight possible colours, depending on the combination of three buttons being pressed as it moves. You can view it as a sort of colour etch-a-sketch.

MOUSE_PAINT needs a three button mouse to allow access to the full range of Mode8 colours, but you can use it in four colours with a two-button Microsoft mouse.

A real mouse painting package would reserve certain areas of the screen for ink post and other controls, rather than dedicating the

buttons to colour control, but the example does show how easily colour mouse drawing can be implemented.

The PD Qdos emulator for the Amiga includes another SuperBasic program that uses the PTR_ extensions. PALETTE_BAS, on the Qdos utilities disk, lets you choose the QL emulator screen colours from a palette of 4096 shades. You use the mouse to select and move sliders on the screen, and the colours change as you do it. Alas, this only works on machines with palette hardware.

Keyboard Emulation

It is always best to tailor programs specifically for the mouse, capable of tracking and responding to precise

mouse movements and button combinations.

Unfortunately most QL programs were written before there was a standard for SuperBasic mouse commands and functions, and it would be good to be able to use them with the mouse even though they were written for keyboard or perhaps joystick control.

That is the main reason for this update to the DIY Mouse project. The extra code in the main listing adds optional emulation of cursors and other major control keys via the mouse. Thus the mouse works with Quill, ED, Devpac, Turbo, Abacus, Spy, Editor, and other cursor-controlled packages.

First you must load and turn on the appropriate code, and set the right baud rate: normally BAUD 1200. The mouse functions will

now respond to movements of the mouse or buttons, but keyboard entry will be unaffected. When you want keyboard emulation, so the cursor tracks mouse movements, issue the new command:

PTR_KEY 1,0

The mouse will now generate cursor key-presses as it moves. Use PTR_INC to set the speed, and PTR_KEY 0,0 to turn it off. You can still use the keyboard as normal. The mouse signals are in addition to joystick or key-presses.

Mouse Wrapping

The second parameter of PTR_KEY is another switch. It allows mouse co-ordinates to wrap from one edge of the screen to the opposite edge.

```

* ***** DIY TOOLKIT - MOUSE/POINTER DRIVER + KEY EMULATION
* This SuperBasic and Qdos key queues to a PC serial mouse
* Version 2.8 updates only, Copyright © 1993 Simon N Goodwin
*
* Extensions over version 1.1:
*
* 2.1 Optional co-ordinate wrap-around at screen borders
* 2.2 PTR_KEY extension to control wrap and key emulation
* 2.3 Optional Keyboard emulation for cursors and buttons
* 2.4 PTR_FN% function to read pointer characteristics
*
qkeys          equ      1          Configure to zero or one
** 2.1 ** Ensure QKEYS is valid (0 or 1; value 1: used later
*
        ifeq qkeys
        ifne qkeys-1
        fail
        endc
        endc
** 2.2 ** Key codes
*
left          equ      192          Cursor arrows
right         equ      200
up            equ      208
down         equ      216
button1       equ      32           Space
button2       equ      16           Enter
button3       equ      27           Escape
*
sv_keyq       equ      76           Key queue system variable
* These variables are relative to PREFIX
*

```

If you set PTR_KEY 0,1 before loading MOUSE_PAINT you will find that the cursor swaps round at the edges of the screen, giving you a quick route to the other side, top or bottom. This 'wrapping' is a popular option available on Apple Macs and other windowing systems. It was easy to do, so I added it for the second version.

PTR_KEY works like all the other commands to set variables, which is why I lazily implemented one command to control both the version 2 improvements - pointer wrap and keyboard emulation.

The parameters are simple switches, with zero meaning off, so PTR_KEY 1,0 turns cursor key emula-

tion on and co-ordinate wrapping off. The order should be easy to remember once you note that the WRAP parameter is on the outside.

PTR_INC

PTR_INC sets the number of mouse pulses that correspond to one move of the cursor in either direction. It was introduced a couple of months ago, but finds its niche in the new version 2 of the DIY Mouse driver, which sends cursor-control characters to tasks normally controlled from the keyboard.

The increment is in Mode4 pixels, and starts off

at 12 in X and 24 in Y. This is less sensitive than the settings for the first version, published in QL World 1993 issue 10, but it works better for me. It depends how far you want to move with one prod of the mouse.

If you use Mode8 or wide Mode4 characters the cursor will respond much faster to horizontal moves than to vertical ones, and it may be worth increasing the X increment, with a command like PTR_INC 24,24. It pays to experiment - some people like slower vertical movement since it reduces the risk of moving off the current line when scanning left and right.

You can use PTR_INC to scale the movement on

screen to suit your mouse-pad. Staying in Mode8 for the time being, try PTR_INC 30,25 for easy character-positioning, and PTR_INC 6,5 if you want fast movement. Double the first value, the X increment, if using Mode4, and double the second value to suit double-height characters.

Main Listing

The main listing this month consists of updates to the DIY Mouse driver source for Version 2. This implements keyboard emulation, so that programs can be controlled with the mouse instead of joystick or cursor keys. The lines have been extracted from the full MOUSE_ASM source, which is available as usual to readers on disk or cartridge.

DIY Toolkit Volume I includes all the new source code, and eight ready-made mouse drivers. The volume is available by post from former Quanta editor Dr. Bill Fuggle. Volume I consists of SuperBasic prototypes, wiring details, example programs, assembler source and binary code for "Mouse Systems PC" and "Microsoft" serial mouse drivers to suit either serial port.

DIY Toolkit volumes cost three pounds each on disk or microdrive cartridge, anywhere in the world, and come with laser-printed documentation if you order two or more. Twenty four volumes are available from DIY Toolkit, 86 Lordswood Road, Harborne, Birmingham B17 9BY. Please make cheques payable to DIY Toolkit, and send a stamped self-addressed envelope if you would like further details.

If you really enjoy typing you can merge the new lines into the source listed in previous issues. The new lines are intended to be added to the first two parts of the listing, and come in three sections. The first section gives names to key-

```

*                                     PTR_INC%
latest_x    equ    0                Current co-ordinates
latest_y    equ    2                "
limit_x     equ    4                Right margin limit
limit_y     equ    6                Top margin limit
step_x      equ    8                Counts per horizontal move
step_y      equ    10               Counts per vertical move
button_bits equ    12               Bits shadow each button
synchro     equ    14               Input byte number, 1 to 3
initial     equ    15               Two-button initial byte
serial_id   equ    16               8+9 zero or serial channel ID
*

drift_x     ifne    qkeys           ** 2.0 ** Accumulated X drift
drift_y     equ    20               Accumulated Y drift for keys
key_flag    equ    22               Set to request key queueing
wrap_flag   equ    26               Set to allow wrap at edges
endc

*
var_000     equ    20+qkeys*8       ** 2.0 ** 8 bytes key space
*
*****\*****
* TWO BUTTON KEYBOARD EMULATION & CO-ORDINATE WRAPPING UPDATES
*
* (1) Replace BPLS with word BPL to BYTE THREE
* (2) Insert BSR BUTTON_KEY before storing D0 at BUTTON_BITS(A4)
* (3) Revised X and Y move handlers:
*
        bsr    horizontal           New line to drive key queue
move.w     latest_x(a4),d0          As before, so far
add.w      d1,d0
bpl.s      enough_x               Check the upper limit
tst.w      wrap_flag(a4)           Are we wrapping?
beq.s      clear_xout
add.w      limit_x(a4),d0          Wrap co-ordinate up
in_xrange  move.w     d0,latest_x(a4)
near_xrange bra      next_byte     Local diversion
*
enough_x   move.w     limit_x(a4),d1
cmp.w      d1,d0                  Over the top?
bhs.s      in_xrange
tst.w      wrap_flag(a4)           Are we wrapping?
beq.s      clip_x
sub.w      d1,d0                  Wrap co-ordinate down
bra.s      in_xrange
clip_x     move.w     d1,d0          Set current Y to limit
bra.s      in_xrange
clear_xout clr.w      d0            Zero
bra.s      near_xrange            Loop via local label
*

```



```

* corresponding changes to the code to process Y deltas
*
        bsr        vertical      Drive the keyboard
        move.w     latest_y(a4),d0
        add.w      d1,d0         D0 is new candidate Y
        bpl.s      enough_y      Check the upper limit
        tst.w      wrap_flag(a4) Are we wrapping?
        beq.s      clear_yout
        add.w      limit_y(a4),d0
        bra.s      in_yrange
*
enough_y    move.w     limit_y(a4),d1
            cmp.w      d1,d0      Over the top?
            bls.s      in_yrange
            tst.w      wrap_flag(a4) Are we wrapping?
            beq.s      clip_y
            sub.w      d1,d0      wrap co-ordinate down
            bra.s      in_yrange
clip_y      move.w     d1,d0      Set current Y to limit
*
in_yrange   move.w     d0,latest_y(a4)
near_sync   bra        clear_sync Get ready for new message
clear_yout  clr.w      latest_y(a4)
            bra.s      near_sync Return via local label
*
*****
*
** 2.0 ** Cursor movement routines; add near end of listing
* On entry D1 contains the new move delta and is preserved
* Uses A4, A6; corrupts A5, A2, D3, D7; IO.QIN hits A3
*
horizontal  tst.w      key_flag(a4)
            beq.s      no_keys    Key entry disabled
            move.l     sv_keyq(a6),d0
            beq.s      no_keys
            move.w     d1,d7      Save delta
            move.l     a2,d6      Save input queue pointer
            movea.l     d0,a2      A2 -> Key queue
            move.w     drift_x(a4),d3
            movea.w     $E0.w,a5    IO.QIN vector
            add.w      d1,d3      D3 is total delta
            bmi.s      go_left
*
            moveq      #right,d1
*
more        sub.w      step_x(a4),d3 Try a move
            bcs.s      add_xout
            jsr        (a5)        Cursor right
            bra.s      more
*
go_left     moveq      #left,d1
less        add.w      step_x(a4),d3 Try a move
            bpl.s      sub_xout
            jsr        (a5)        Cursor left
            bra.s      less
*
sub_xout    sub.w      step_x(a4),d3
            bra.s      set_xdrift Store remainder
add_xout    add.w      step_x(a4),d3
set_xdrift  move.w     d3,drift_x(a4)
            move.l     d6,a2      Restore serial queue
            move.w     d7,d1      Restore delta
no_keys     rts
*
vertical    tst.w      key_flag(a4)
            beq.s      no_keys
            move.l     sv_keyq(a6),d0
            beq.s      no_keys    No queue to be stuffed
            move.w     d1,d7
*
            ifeq      buttons-3
            neg.w      d1          Reverse Y direction
            endc
*
            move.l     a2,d6      Save SER queue address
            move.l     d0,a2      A2 -> Key queue
            movea.w     $E0.w,a5    A5 = IO.QIN vector
            move.w     drift_y(a4),d3
            add.w      d1,d3      D3 is total delta
            bmi.s      go_up
            moveq      #down,d1
descend     sub.w      step_y(a4),d3 Try a move
            bcs.s      add_yout    Don't go too far
            jsr        (a5)
            bra.s      descend

```

board codes and variable offsets in the interrupt linkage.

These variables are private to the handler and not directly available to Basic, but there are lots of interesting things among the interrupt handler's variables. It seems a good idea to have some way to read them all, so I have added a routine to do this in the DIY Toolkit Volume. The assembly code is similar to that for BUT-TON%, but the parameter is a word number to be used as an offset when reading.

PTR_FN% can read any word of the DIY Mouse variables. It takes one parameter, the offset of the required word, starting at zero (LAT-EST_X), so PTR_FN%(2) reads the right margin limit (the third word) PTR_FN%(4) reads STEP_X and so on. Parameters for PTR_FN% are tabulated near the start of the new assembler listing.

Conditional Code

The single file MOUSE.ASM can create eight different versions of the driver, so it uses conditional assembly. The lines between an IFEQ or IFNE and the matching ENDC are only included in the assembled code if the expression after the IF is zero (IFEQ) or non-zero (IFNE). If your assembler does not support conditional assembly you need to comment out the lines that you do not need.

Set QKEYS to one or zero depending on whether or not you want extra code to give the option of keyboard emulation. You do not need this for 'mouse aware' programs that use extensions like PTR_X% and BUT-TON% to read the mouse directly and precisely.

If QKEYS is one the mouse driver has the ability to send cursor movement characters to the current task, so you can move the cursor with a mouse instead of the arrow keys. The sub-

```

*
go_up      moveq    #up,d1
asc_end    add.w     step_y(a4),d3    Try a move
          bpl.s     sub_yout
          jsr      (a5)
          bra.s     asc_end

*
sub_yout    sub.w     step_y(a4),d3
          bra.s     set_ydrift
add_yout    add.w     step_y(a4),d3
set_ydrift  move.w     d3,drift_y(a4)
          move.l     d6,a2            Restore serial queue
          move.w     d7,d1            Restore new delta
          rts

*
* Routine to translate buttons to key-codes; new buttons in D0
*
button_key  tst.w     d0                Any action?
          beq.s     wits_end
          tst.w     key_flag(a4)      Key emulation on?
          beq.s     wits_end
          move.l     sv_keyq(a6),d1   Is there a key queue?

          beq.s     wits_end            No queue now so ignore
          move.w     d0,d7
          move.l     a2,d6            Save -> Serial queue
          move.l     d1,a2            A2 -> Key queue
          movea.w     $E0.w,a5        A5 = IO.QIN vector
          move.w     button_bits(a4),d1

          btst      #0,d0
          beq.s     not1
          btst      #0,d1            Already down?
          bne.s     not1
          moveq      #button1,d1      Send new keypress
          jsr      (a5)
          ignore 'no room' error
not1        btst      #1,d0            Button two?
          beq.s     not2
          btst      #1,d1            Already down?
          bne.s     not2
          moveq      #button2,d1      Queue it
          jsr      (a5)

*
not2        ifeq      buttons-3
          btst      #2,d0            Third button?
          beq.s     restore
          btst      #2,d1
          bne.s     restore
          moveq      #button3,d1      Try to queue it
          jsr      (a5)
          endc

*
restore     move.w     d7,d0            Restore new buttons
          movea.l     d6,a2            Restore input queue
wits_end    rts

```

sequent IFNE and ENDC directives ensure that an error occurs if QKEYS is set incorrectly.

The second part of the assembler listing is marked off by a row of stars. It is a re-write of the two button Microsoft mouse driver, extended to allow co-ordinate wrapping and keyboard emulation.

The keyboard emulation and co-ordinate wrapping code was added by re-writing the inner loop of the mouse handler, adding calls to three new subroutines HORIZONTAL, VERTICAL and BUTTON_KEY which issue key-presses as required.

If the version 2 variable WRAP_FLAG is set then an overflow or underflow wraps the co-ordinate to the opposite edge of the screen. I decided it was better to copy and amend the four routines for each axis and mouse-type than to try to put conditional code in each one.

The tweaks may appear complicated but should be reliable; I set out to introduce the smallest number of changes and new lines that would do the job simply. It would be possible to save bytes by merging the vertical and horizontal keyboard routines, but this could introduce new bugs.

The final section should be added before the names and addresses at the end of the assembly file. It contains code to emulate key-presses by putting characters into the current key-buffer as the mouse moves.

The mouse buttons also generate key-codes, corresponding to SPACE, ENTER and ESCAPE (if you use a Mouse Systems compatible pointing device with three buttons). These are the main keys for QRAM and most pointer-based programs.

The code for PTR_KEY is not listed here as it is exactly the same as that for PTR_MAX, PTR_INC and their ilk, except that it adds

24 to A4 - the offset of KEY_FLAG - after calling FIND_POS and before returning via SETWORDS.

Performance

I have tested all these programs with an AXELEN dual-standard switchable mouse, which cost £10 at an All-Formats Computer Fair. The middle button is ignored when it is switched to Microsoft mode. The DIY Toolkit drivers have also been tested with a similar mouse supplied by Dennis Briggs.

Serial mice are widely available at low prices and come with a nine or 15 pin PC connector. If you own a British-made QL you need to replace this with a six-pin plug to suit your SER port. Samsung QLs have a nine-pin connector which may accept the PC plug, but you'll still need to check that the internal links match your chosen QL port. The necessary connections were explained on page 20 of QL World 1993 issue II.9.

Dennis Briggs of Adman Services can build you an adapter for £4 if you do not feel like making up your own. He can also supply PC mice complete with the required QL adapter and the full DIY software Volume for £15 inclusive. Contact Dennis Briggs at Adman Services, 53 Gilpin Road, Admaston, Telford, Shropshire, England, or call him on (0952) 255895.

The DIY mouse handlers check the format of the first byte of each message. If it is implausible they discard it and wait for a valid one. This means that the handler re-synchronises itself at the start of the next complete message.

If you are using a switchable joystick with Sinclair's serial input chip, you will probably find that the two-button setting is the most reliable. Three-button signals are more likely to lose synchronisation if serial input

```

100 REMARK MODE 8 mini MOUSE_PAINT
110 REMARK Use combinations of
120 REMARK 3 buttons to pick 8
130 REMARK colours for drawing
140 REMARK Uses DIY PLOT + DRAW exts
150 :
160 BAUD 1200
170 PTR_ON
180 REMARK Full screen
190 OPEN #3,scr_512x256a0x0
200 PTR_MAX 511,255
210 PTR_POS 0,0
220 PTR_KEY 0,0 :REMARK Wrap?
230 MODE 8
240 x%=0 : y%=0
250 CLS #3
260 BLINK (x%),(y%)
270 REPEAT pain
280   oldx%=x% : oldy%=y%
290   x%=PTR_X% : y%=PTR_Y%
300   BLINK (oldx%),(oldy%)
310   INK #3,BUTTON%(0)
320   PLOT #3,x%,y%
330   BLINK (x%),(y%)
340   AND REPEAT pain
350   CLOSE #3
360 :
370 DEFINE PROCEDURE BLINK(a,b)
380 REMARK Non-destructive '+' cursor
390 OVER #3,-1 : INK #3,7
400 a=a-2 : IF a<0 : a=0
410 IF a>507 : a=507
420 b=b-2 : IF b<0 : b=0
430 IF b>251 : b=251
440 PLOT #3,a+2,b : DRAW #3,a+2,b+4
450 PLOT #3,a,b+2 : DRAW #3,a+4,b+2
460 OVER #3,-1
470 END DEFINE BLINK

```

gets lost. The best way to avoid such problems is to upgrade your system with the Hermes chip, which dramatically improves the performance of the QL serial inputs.

The machine-code versions read the serial input queue directly, but can still lose step if Sinclair's old IPC drops bytes on their way from the mouse to Qdos. This is most likely if the QL is busy with a device and cannot read serial bytes for a while. They pile up in the 8049 co-processor, and soon overflow.

Checklist

If your QL is otherwise fine but your serial mouse won't work, check these things first:

(1) Did you load the correct code file for your mouse and adapter? (2) Is the mouse plugged into the correct port for the adapter? (3) Have you set BAUD 1200, PTR_ON, and PTR_KEY 1,1 if

needed?

If the mouse is a switchable one:

(4) Is the mouse switched to the right protocol (PC for three buttons or MS for two)?

If the answer to any of these questions is no, reset the QL, check the connections and settings and try again.

If it still won't work, try the mouse on a PC or another QL system. If it works there, but it is erratic on your QL,

you need Hermes to fix your QL serial input. This is more likely with old three button mice than with the two-button Microsoft breed.

In either case, **Hermes** is an excellent QL upgrade, ideal for this purpose. It has proper serial inputs, written by a Bulletin board boss who knows RS-232 inside out, so it goes much faster and suits all sorts of serial devices.

A New Mousetrap?

Well, that completes the biggest DIY Toolkit project to date - We've ended up with over seven hundred lines of code, eight different mouse drivers, four SuperBasic demonstrations and six Quill files of documentation. DIY Toolkit Volume I also includes a Turbo-compiled mouse driver with keyboard emulation that works fine with Hermes.

I plan to tackle something a little simpler in the next DIY Toolkit project; as ever I

need your suggestions for the column. Readers have sent in dozens of good ideas over the years, and I've implemented most of them. If there's anything new you'd like to see in SuperBasic, please let me know the details and I'll write it for you, and the other DIY Toolkit enthusiasts.

Update

I've made a couple of improvements to the INPUT\$ and SET_POS code in QL World 1993 II.8. The DIY Volume has been updated and I want to share the new code with people who type in the assembly listings straight off the page.

The INPUT\$ function upsets Turbo and Supercharge because it does not set A1 on exit. The compilers use the register value, rather than BV.RIP, to find the result. Section 10.7.5 of Adrian Dickens' QL Advanced User Guide says that both A1 and BV.RIP should be set appropriately, but QL roms only look at BV.RIP, while the compilers use A1.

I didn't spot this for a while as interpreter tests worked fine and Turbo comes with its own INPUT\$ function. To correct the assembly code, add the line MOVEAL D5,A1 after the line annotated SET BV.RIP, near the end of the code for INPUT\$.

My article said that SET_POS goes to the end of a file without error if asked to move to a position past the end. This feature was not used in the accompanying SuperBasic, and it seems I forgot it, as users are getting an End Of File report. This code traps any EOF error after the TRAP #3:

```

TRAP_EOF CMP.W #-
10,D0
BNE.S RETURN
MOVEQ #0,D0
RETURN RTS

```

Mea Culpa! Thank you for alerting me to this mistake.

Machine Code for Beginners

In part 7, Alan Bridewell introduces Trap Calls.

In part 6 we started to look at how we can make our machine code routines do the kind of things we really need them to do: opening channels, writing to the screen, and so on. What

we do not do is to actually write the code to do this ourselves because the code already exists in the QL's rom. All we need to do is to access this code and incorporate in our routine

Rom Routines

The routines in the rom are of two types, called "vectored utilities" and "trap calls". In part 6 we looked at vectored utili-

ties, so now we shall move on to trap calls. As far as running your programs is concerned there is no noticeable difference between the two types of routine. If you have the facility to trace through your machine code, one instruction at a time, you will notice one difference. When you use a vectored utility, you will trace through the whole subroutine as though it was part of your code. When you use a trap call, it will not be traced.

As far as you, as programmer, are concerned, the big difference between vectored utilities and trap calls is the way you access them in your program. However, we should look at what exactly a trap call is, although much of the detail of this is very hard going, and not really needed unless you intend to do some high powered programming. I shall only cover it in outline.

The microprocessor in the QL can operate in two modes, called "user mode" and "supervisor mode". User mode is what programs normally

use, and everything mentioned in this series of articles is in user mode. When the processor is in supervisor mode, the program in user mode is temporarily suspended, while special supervisor mode routines are run. These routines can be exactly the same as user mode routines, but can also contain some special "privileged" instructions.

Supervisor Mode

Much of Qdos works in supervisor mode, and users would not normally want to write supervisor mode code unless they were trying to get Qdos to do something it was not designed for. This is usually restricted to people writing software to run hardware which Qdos cannot operate without some extensions, for example, disc drives. This kind of programming is well outside the scope of these articles.

Trap calls are Qdos routines which run in supervisor mode. The TRAP instruction allows the user mode program to access the supervisor mode routines in a very controlled way, which prevents you causing something nasty to happen by mistake. It allows you to input whatever parameters you wish into the routine, and then the supervisor mode routine tries them out. If it can make sense of them, it will carry them out, doing whatever the routine is supposed to do, before returning to user mode at the next instruction in your code, with no error code.

If it cannot make sense of them, it will also return to user mode at the next instruction in your code, but this time returning with an error code in register D0. What this means is that your mistake will not crash the machine, as long as you take appropriate action on receiving the error code.

16 Calls

The microprocessor recognises 16 different trap calls, numbered #0 to #15, but the QL only uses the first five, #0 to #4. Trap #0 is concerned with going into supervisor mode, and unless you are an expert or are intent on crashing your machine and maybe

```

1
2 *****
3 *****
4 *****
5 *****
6 *****
7 *****
8 *****
9 *****
10 *****
11 *****
12 *****
13 *****
14 *****
15 *****
16 *****
17 *****
18 *****
19 *****
20 *****
21 *****
22 *****
23 *****
24 *****
25 *****
26 *****
27 *****
28 *****
29 *****
30 *****
31 *****
32 *****
33 *****
34 *****
35 *****
36 *****
37 *****
38 *****
39 *****
40 *****
41 *****
42 *****
43 *****
44 *****
45 *****
46 *****
47 *****
48 *****
49 *****
50 *****
51 *****
52 *****
53 *****
54 *****
55 *****
56 *****
57 *****
58 *****
59 *****
60 *****
61 *****
62 *****
63 *****
64 *****
65 *****
66 *****
67 *****
68 *****
69 *****
70 *****
71 *****
72 *****
73 *****
74 *****
75 *****
76 *****
77 *****
78 *****
79 *****
80 *****
81 *****
82 *****
83 *****
84 *****
85 *****
86 *****
87 *****
88 *****
89 *****
90 *****
91 *****
92 *****
93 *****
94 *****
95 *****
96 *****
97 *****
98 *****
99 *****
100 *****

```

even damaging your hardware, you should leave well alone. I shall not mention it again.

Trap #1 is the manager trap. It is concerned with all the memory allocation, priority of

mutitasking jobs, keyboard access, clock, and various other bits and pieces that can be grouped under the name "management".

Trap #2 is the input and out-

put allocation trap. Its work involves formatting microdrives and discs, opening and closing channels, and deleting files.

Trap #3 is the input and out-

put utilisation trap. Having a channel open already, this trap can do whatever can be done to the channel. This will, of course, depend on what type of channel it is, a screen or console, a new file to be written to, an old file to be read, or maybe overwritten. Most of the interesting things you might want to do are done by trap #3 calls.

Trap #4 is concerned with making adjustments for certain addressing modes when using SuperBasic. It need not concern us here.

So how do we use these trap calls? The method is very similar to using the vectored utilities mentioned in part 6. First we must make sure all the necessary parameters are in the correct registers, and any data or parameter tables have been correctly set up. We then have to make sure the trap call knows which we want out of the various things it can do. We do this by entering the correct number into register D0. Finally, we use the instruction TRAP with the appropriate operand (#0, #1, #2, #3 or #4).

Open and Close

To show how this works, Listing 1 is a routine which uses trap #2 to open and close a file, and trap #3 to input text to a window, and load a file to the screen. It does the same thing you can do from SuperBasic with the LBYTES command using the screen ram address. In order to make sense of the rest of this article, you will probably have to read it alongside Listing One.

It starts by opening a console channel with the vectored utility used in part 6. The difference in this case is that the parameter table has already been set up to give the required window, so it does not require the CALL from SuperBasic

to give the parameters. (This is the way the utility would normally be used. It was only done that way in part 6 to enable experimentation.) As before, we shall store the console channel I.D.

Next, we need a prompt in the window. This involves writing some text to the window in exactly the same way as in part 6, so this needs no further explanation.

```

100 z=RESPR(512)
110 LBYTES flp2_LISTING1 code,z
120 CLS#0:CLS#1:CLS#2
130 PRINT#0, "After each screen is loaded, you can press <SPACE> to prompt for another file
140 PRINT#0, "or <ESC> to quit the program."
150 PRINT#0, "Press <ENTER> to continue...."
160 REPEAT loop
170 IF CODE(INKEY$(#0,-1))=12 THEN EXIT loop
180 END REPEAT loop
190 REPEAT main
200 CALL z
210 REPEAT loop2
220 key = CODE(INKEY$(#0,-1))
230 SELECT ON key
240 = 32: EXIT loop2
250 = 27: STOP
260 END SELECT
270 END REPEAT loop2
280 END REPEAT main

```

Listing 3

```

100 REMark Sinclair QL World HEX LOADER v 3
110 REMark by Marcus Jeffery & Shaun N Goodwin
120
130 CLS :RESTORE :K&A: space start=PE$chk$space)
140 PRINT "Loading hex file HEX_LOAD start"
150 IN$=IN$ : Save to file : "F3"
160 SBYTES 15,start,byte-STOP
170
180 DEFINE FUNCTION DECIMAL(x)
190 RETURN CODE(h$(x))-48-7*(h$(x)>"9")
200 END DEFINE DECIMAL
210
220 DEFINE PROCEDURE HEX_LOAD(start)
230 byte=0:checksum=a
240 REPEAT load_hex_digits
250   REA: h$
260   IF h$="" THEN EXIT load_hex_digits
270   IF ISN(h$) MOD 2
280     PRINT "Out number or hex digits in: ",n3
290     STOP
300   END IF
310   FOR b=1 TO LEN(h$) STEP 2
320     h$=DECIMAL(h$(b)).b=DECIMAL(h$(b+1))
330     IF h$b<0 OR h$b>15 OR h$b<0 OR h$b>15
340       PRINT "Illegal hex digit in: ",h$.STOP
350     END IF
360     POKE start+byte,16*h$b+b
370     checksum=checksum+16*h$b+b
380     byte=byte+1
390   END FOR b
400 END REPEAT load_hex_digits
410 READ check
420 IF check<>checksum
430   PRINT "Checksum incorrect. Recheck data. ":STOP
440 END IF
450 PRINT "Checksum correct. Data entered at: ",start
460 END DEFINE HEX_LOAD
470
480 REMark Space requirements for the machine code
490 DATA 172
500
510 DATA "49FA0086":REMARK
520 DATA "547800C8":REMARK
530 DATA "4F82":REMARK
540 DATA "43FA0088":REMARK
550 DATA "2288":REMARK
560 DATA "43FA0082":REMARK PTEXT
570 DATA "2051":REMARK
580 DATA "49FA0084":REMARK
590 DATA "547800D0":REMARK
600 DATA "4E92":REMARK
610 DATA "43FA0072":REMARK
620 DATA "2051":REMARK
630 DATA "43FA0084":REMARK
640 DATA "7002":REMARK
650 DATA "543C0064":REMARK
660 DATA "75FF":REMARK
670 DATA "4E43":REMARK
680 DATA "5081":REMARK
690 DATA "41FA0072":REMARK
700 DATA "3081":REMARK
710 DATA "41FA006C":REMARK
720 DATA "7601":REMARK
730 DATA "72FF":REMARK
740 DATA "7001":REMARK
750 DATA "4E42":REMARK
760 DATA "4A80":REMARK
770 DATA "670A":REMARK
780 DATA "547800CA":REMARK
790 DATA "4E92":REMARK
800 DATA "4EFAFEBC":REMARK
810 DATA "43FA0040":REMARK GOT_FILE
820 DATA "2288":REMARK
830 DATA "243C00000000":REMARK
840 DATA "227C00020000":REMARK
850 DATA "76FF":REMARK
860 DATA "207A002C":REMARK
870 DATA "7048":REMARK
880 DATA "4E43":REMARK
890 DATA "43FA0024":REMARK
900 DATA "2051":REMARK
910 DATA "7002":REMARK
920 DATA "4E42":REMARK
930 DATA "43FA0016":REMARK
940 DATA "2051":REMARK
950 DATA "7002":REMARK
960 DATA "4E42":REMARK
970 DATA "4E75":REMARK
980 DATA "0203":REMARK CON
990 DATA "0402":REMARK
1000 DATA "0200":REMARK
1010 DATA "0010":REMARK
1020 DATA "0000":REMARK
1030 DATA "0000":REMARK
1040 DATA "00000000":REMARK ID
1050 DATA "00000000":REMARK FILE
1060 DATA "000C":REMARK TEXT
1070 DATA "46494C452045414D45203F20":REMARK DC.B
1080 DATA " ",15029

```

The program now needs to input the name of the file by fetching a line of characters from the keyboard. This is part of input/output utilisation and involves a trap #3 call. Before the trap call can do this it needs the channel ID of the console channel in register A0. It also needs the address of a buffer to store the text in register A1 with the length of the buffer in register D2. In order to make sure the characters going into the buffer do not overwrite your program or parameters, the buffer should come at the end.

In order that the trap call knows which bit of input/output utilisation it is to do, it needs a number in register D0 to tell it. The number for fetching a line of text is #2. (This number is usually given the label IO_FLIN, and your assembler may accept this label instead of the number.)

Finally, trap #3 requires a "timeout" in register D3. This is the time, in fiftieths of a second, the call will wait until the utilisation is completed. It will also accept zero to mean "don't wait at all" and -1 to mean "wait for ever if necessary". Frankly, I have always found a timeout of -1 works fine, and that is what we shall use.

With all the parameters correctly positioned in the registers, we make the trap call with the line

TRAP #3

At this stage, we could check register D0 for an error code. Apart from an error in typing in the listing, the only possible error is a file name which is too long for the buffer, which would give a "buffer overflow" error. Since we have made the buffer 100 bytes long, this is very unlikely to happen. I shall come back to this point later.

Juggling

With a successful trap call, the buffer should now contain the name of the screen file we wish to open. We now have to use that name to actually open the file. However, in order to do this, we need to do a little juggling, because the buffer does not contain the file name in quite the correct form.

The trap #3 call will leave the file name in the buffer. But the string of characters will contain the line feed character we get when we press Enter at the end of the line. It also leaves the length of the string (including the line feed) in register D1.

Now the trap #2 call we will use to open the file requires the file name in a slightly different form. It expects the file name to begin with a word (two bytes) which give the number of characters in the file name, followed by the file name itself, but without the line feed character at the end. To get over this difference, we reserve a word (two bytes) immediately before the buffer. We have labelled this word BUFPOS. We subtract one from the character count in register D1, and then move the resulting number into BUFPOS. If we now use BUFPOS rather than BUFFER as the name of the file to be opened, it will be in the correct form, starting with the word giving the number of characters, followed by the characters themselves. Because we have subtracted one from the character count, it will ignore the line feed character at the end.

The next step is to open the file with the trap #2 call. Before we can do this we need to put all the appropriate data into the registers. Register A0

must contain the address of the file name, BUFPOS. Register D3 must contain a number depending on what kind of file is to be opened. The file is an old one, and we may be sharing it (it may also be read by another multitasking program at the same time), and this requires a one in register D3.

Job ID

Register D1 requires the job ID of the job using the channel. It is possible for one multitasking job to open a file for another, in which case, the appropriate job ID would need to be found. However, the trap call will accept -1 in D1 to mean "this job".

Finally, it needs a one in register D1 to tell it open the file. The number one has the label IO_OPEN, which your assembler may accept in place of one. We can now make the trap call with the instruction

TRAP #2

This time we must check for an error after returning from the trap call. Even if our code is correct (and it should be) there is the very good chance that the file name typed in is incorrect so that no channel has been opened, in which case we need to take appropriate action.

There is a general rule here. If the parameters are written into the program, there should be no error, and so no need to check. But if any of the parameters come from user input, an error is possible and should be checked for.

In part 6 we tested for an error return by comparing register D0 with zero using the instruction

```
CMPI.W #0,D0
```

CMPI is a general instruction for finding out if we have any particular value, or a bigger or smaller value. But when we do error testing, we are only interested in whether we have a zero or a negative number. For this we have another instruction, TST, which is rather more concise, and so runs faster. Normally, we should use TST for error testing. I only used CMPI before because it is a very useful

instruction in many situations, and is well worth being familiar with.

What TST does is simply tell us if the number we have is zero or negative by setting the appropriate flag in the status register. This is exactly what we need here.

Branch and Hang?

After the TST instruction, a BEQ instruction will branch if the zero flag is set (ie no error) to the next bit of code when everything goes to plan. In our program this has the label GOTFILE. But what do we do if an error is detected, and the program goes to the next instruction, instead of branching to GOTFILE? We could simply abort the program and return to SuperBasic with a RTS instruction. This would not crash the QL, but it would drop you out of the program, which might be just as annoying. There is a much better alternative.

There are really only two possibilities for the error. It could have been an error in typing in the file name, or it could be that the file you want is not actually on the medium in that particular drive. Either way, the best thing to do is to print an appropriate error message on the screen, then loop back to allow you to retype the file name. This is what the next three lines do.

When the file is successfully opened, the file channel I.D. is left in register A0, so we store it in a space reserved for it with the label FILE.

The next job is to load the file into the screen ram with a trap #3 call. To do this we need the file length in register D2. If it is a normal screen file it will be 32K long, which is \$8000 in hex. The address to start loading must be put in register A1, and this is the start address of the screen RAM, \$20000. To indicate an infinite timeout we put -1 in register D3. To show that what we are doing is loading a file, we put \$48 in register D0. (\$48 has the label FS_LOAD which your assembler might accept in place of \$48.) This is finally followed by the trap call with the instruction

TRAP #3

If all has gone well, your saved screen file should load into the screen ram and appear on your screen.

At this point we should tidy things up by closing the channels we have now finished with, that is, the file channel and the console window channel. This is done in each case by loading the channel ID into register A0, loading #2 into register D0 (#2 has the label IO_CLOSE which your assembler may accept) and then making the trap #2 call. After all this we return to SuperBasic with RTS.

After the code come all the various spaces reserved for parameters and data. These can come in any order, except that the BUFFER should come last (and, of course, BUFPOS must come immediately before it.)

Listings

Listing Two is a small SuperBasic routine to run the code. It needs no explanation. Listing 3 is the code contained in Marcus' and Simon's Hex Loader for those who do not have an assembler.

Having got the thing to work, the next job is to experiment with it to show that you really do understand what is going on. You could do simple things like changing the window parameters, or changing the prompt, but that has no impact on the trap calls. There is a much more obvious improvement you could try.

As the program stands, you must eventually give it a usable screen file name, otherwise it will repeat the prompt forever, or crash the system when you try to load the wrong file type. Either way, if you can't give it a usable file name, you will have to reset the QL to get back to SuperBasic.

One way out of this is to test for a zero string length before trying to open the file, and jumping to a RTS instruction if a zero string is found. This will enable you to leave the program by pressing Enter at the prompt without having to give a file name. This will make it a lot more user friendly.

Subtle ...

For something more subtle you can look into this. In this program we have religiously saved each channel ID and loaded the channel ID back into the registers when they are needed. While in general this process is necessary, if we are using a channel immediately it is opened, the channel ID is already in the register and does not need reloading. Also, if we are not going to need it again later, there is no need to store it. It is an interesting exercise to see how many redundant lines of storing and reloading channel IDs can be removed and still allow the program to run. If you can do this successfully, then you probably have a good grasp of how the program works.

This article only gives a glimpse of what can be done with trap calls. To go further would need long lists of all the things the trap calls do and what needs to be in each register for each operation. I have already mentioned published works which contain this information. I would simply add to this list my own series of articles "Systematic M/C Programming", which were printed in QL World from October 1991 to September 1992. They contain a lot of examples of the most useful trap calls, particularly the article in February 1992, which covered most of the trap #3 calls.

Happy coding!

Machine Code part 7

* QL WORLD DIY TOOLKIT - MOUSE/POINTER DRIVER + KEY EMULATION
 * Links SuperBASIC and Qdos key queues to a PC serial mouse
 * Version 2.8 updates only, Copyright © 1993 Simon N Goodwin
 *

* Extensions over version 1.n:
 *

* 2.2 Optional co-ordinate wrap-around at screen borders
 * 2.4 PTR_KEY extension to control wrap and key emulation
 * 2.7 Optional Keyboard emulation for cursors and buttons
 * 2.8 PTR_FN% function to read pointer characteristics
 *

qkeys equ 1 Configure to zero or one
 *

** 2.0 ** Ensure QKEYS is valid (0 or 1); value is used later
 *

```

      ifne    qkeys
      ifne    qkeys-1
      fail
      endc
      endc
  
```

QKEYS 0 or 1 = key emulation

*
 ** 2.0 ** Key codes
 *

```

left      equ    192      Cursor arrows
right     equ    200
up        equ    208
down      equ    216
button1   equ    32      Space
button2   equ    10      Enter
button3   equ    27      Escape
  
```

sv_keyq equ 76 Key queue system variable
 *

* These variables are relative to PREFIX
 *

```

      PTR_FN%
latest_x  equ    0        0    Current co-ordinates
latest_y  equ    2        1
limit_x   equ    4        2    Right margin limit
limit_y   equ    6        3    Top margin limit
step_x    equ    8        4    Counts per horizontal move
step_y    equ    10       5    Counts per vertical move
button_bits equ    12      6    Bits shadow each button
synchro   equ    14       7    Input byte number, 1 to 3
initial   equ    15
serial_id equ    16      8+9    Zero or serial channel ID
  
```

```

      ifne    qkeys
drift_x   equ    20      10    ** 2.0 ** Accumulated X drift
drift_y   equ    22      11    Accumulated Y drift for keys
key_flag  equ    24      12    Set to request key queueing
wrap_flag equ    26      13    Set to allow wrap at edges
      endc
  
```

var_end equ 20+qkeys*8 ** 2.0 ** 8 bytes key space
 *

* TWO BUTTON KEYBOARD EMULATION & CO-ORDINATE WRAPPING UPDATES
 *

* (1) Replace BPL.S with word BPL to BYTE_THREE
 * (2) Insert BSR BUTTON_KEY before storing D0 at BUTTON_BITS(A4)
 * (3) Revised X and Y move handlers:
 *

```

; *****
; OPEN A CON CHANNEL
; *****
.OPEN          LEA.L      CON,A1      ; CON TABLE POINTER IN A1
                MOVEA.W   $C6,A2      ; UT_CON VECTOR IN A2
                JSR        (A2)        ; OPEN CONSOLE CHANNEL
.STORE         LEA.L      ID,A1        ; CHANNEL ID STORE IN A1
                MOVE.L     AO,(A1)     ; STORE CHANNEL ID

; *****
; WRITE TEXT IN CON WINDOW
; *****
.PTEXT         LEA.L      ID,A1        ; CHANNEL ID STORE IN A1
                MOVEA.L    (A1),AO     ; CHANNEL ID IN AO
                LEA.L      TEXT,A1     ; BASE OF TEXT IN A1
                MOVEA.W    $D0,A2     ; UT_MTEXT VECTOR IN A2
                JSR        (A2)        ; PRINT TEXT

; *****
; FETCH A LINE OF CHARACTERS
; *****
.FETCH         LEA.L      ID,A1        ; CHANNEL ID STORE IN A1
                MOVEA.L    (A1),AO     ; CHANNEL ID IN AO
                LEA.L      BUFFER,A1   ; BUFFER ADDRESS IN A1
                MOVEQ      #2,D0       ; #IO_FLINE IN D0
                MOVE.W     #BUFLen,D2  ; BUFFER LENGTH IN D2
                MOVEQ      #-1,D3      ; INFINITE TIMEOUT
                TRAP        #3

; THIS LEAVES THE LENGTH OF THE STRING (INCLUDING THE LF) IN D1.
; IF WE WANT TO USE THIS STRING AS A CHANNEL NAME, WE NEED THE LENGTH OF
; THE STRING (WITHOUT THE LF) JUST BEFORE THE STRING ITSELF (WHICH IS NOW
; STORED IN THE BUFFER). SO:-
; SUBTRACT 1 FROM THE STRING LENGTH....
;
                SUBQ.L      #1,D1       ; SUBTRACT 1 FROM D1
;
; ....AND PUT THE RESULT IN BUFPOS, JUST IN FRONT OF BUFFER
                LEA.L      BUFPOS,A0   ; LOAD BUFPOS IN AO
                MOVE.W     D1,(AO)     ; PUT CONTENTS OF D1 IN BUFPOS

; *****
; OPEN THE FILE
; *****
                LEA.L      BUFPOS,A0   ; BUFPOS IN AO
                MOVEQ      #1,D3       ; OPEN AND OLD, SHARED FILE
                MOVEQ      #-1,D1      ; -1 IN D1 MEANS 'THIS JOB'
                MOVEQ      #1,D0       ; #IO_OPEN IN D0
                TRAP        #2
                TST.L      D0          ; IS THERE AN ERROR?
                BEQ.S       GOT_FILE   ; IF NOT, CONTINUE..

; ...ELSE...

                MOVEA.W    $CA,A2      ; ERRO IN A2
                JSR        (A2)        ; PRINT ERROR MESSAGE IN #0
                JMP        PTEXT       ; REPEAT PROMPT

; STORE FILE CHANNEL I.D WHICH IS NOW IN AO

```



```

.GOT_FILE      LEA.L      FILE,A1      ; ADDRESS OF FILE IN A1
                MOVE.L    AO,(A1)      ; CONTENTS OF AO INTO ADDRESS IN A1

; NOW LOAD THE FILE INTO THE SCREEN RAM

                MOVE.L     #$8000,D2    ; FILE LENGTH IN D2
                MOVEA.L    #$20000,A1   ; LOAD ADDRESS IN A1
                MOVEQ      #-1,D3       ; INFINITE TIMEOUT
                MOVEA.L    FILE,A0      ; FILE CHANNEL I.F IN A0
                MOVEQ      #$48,D0      ; FS_LOAD INDO
                TRAP        #3

*****
                CLOSE THE CHANNELS
*****

                LEA.L      FILE,A1      ; ADDRESS OF FILE CHANNEL I.D.
                MOVEA.L    (A1),AO      ; INTO AO
                MOVEQ      #2,D0        ; IO_CLOSE IN D0
                TRAP        #2          ; CLOSE FILE
                LEA.L      ID,A1        ; ADDRESS OF CONSOLE CHANNEL I.D.
                MOVEA.L    (A1),AO      ; INTO AO
                MOVEQ      #2,D0        ; IO_CLOSE IN D0
                TRAP        #2          ; CLOSE CONSOLE
                RTS              ; RETURN TO SUPERBASIC

*****

SPACE RESERVED FOR CONSOLE CHANNEL PARAMETERS

.CON           DC.W        $0203      ; BORDER COLOUR & WIDTH
                DC.W        $0402      ; PAPER & INK COLOUR
                DC.W        $0200      ; WIDTH
                DC.W        $0010      ; HEIGHT
                DC.W        $0000      ; X ORIGIN
                DC.W        $0000      ; Y ORIGIN

; SPACE RESERVED FOR CONSOLE CHANNEL I.D.

.ID            DC.L        $00000000  ; CONSOLE CHANNEL ID

; SPACE RESERVED FOR FILE CHANNEL I.D.

.FILE         DC.L        $00000000  ; FILE CHANNEL I.D.

; TEXT TO BE WRITTEN IN THE WINDOW

.TEXT         DC.W        $0C         ; NUMBER OF CHARACTERS
                DC.B        "FILE NAME ? " ; CHARACTERS

; BUFLN
.BUFLN        EQU         100         ; LENGTH OF INPUT BUFFER

; BUFFPOS
.BUFFPOS      DC.W        0

; BUFFER
.BUFFER       EQU         *

*****

```

Make sure it
comes out dark
10/1/80

DJC

Dilwyn Jones Computing

41 BRO EMRYS, TAL-Y-BONT, BANGOR,
GWYNEDD, LL57 3YT, GREAT BRITAIN
FAX/TEL: (0248) 354023

DATABASE SOFTWARE

COCKTAILS WAITERS 10 00	
DATA DESIGN 3	£60.00
DATA DESIGN API	£20.00
DBEASY	£15.00
DBPROGS	£15.00
FLASHBACK	£25.00
FLASHBACK SE	£40.00
SUPER DISK INDEXER	£12.00

DTP & CLIPART

PAGE DESIGNER 3	£40.00
Available at last!	
UPGRADE FROM PD2	£25.00
SCANNED CLIPART 1 (U)	£10.00
SCANNED CLIPART 2 (U)	£10.00
THE CLIPART (U)	£12.00

FILE HANDLING

FILEMASTER	£12.00
FILES 2 (U)	£12.00
4MATTER & LOCKSMITHE	£23.50
LOCKSMITHE (U)	£14.95
MDV TOOLCHEST	£14.95
THE GOPHER (U)	£12.00
WINBACK 2	£25.00

FILE TRANSFER

CONVERT-PCX	£10.00
DISCOVER	£20.00
MULTI-DISCOVER	£30.00
OPD INTERCHANGE	£15.00
QL-PC FILESERVER	£24.50
STOQL	£24.50
TEXTIDY	£15.00

GAMES & LEISURE

FIVE GAME PACK (U)	£12.50
FLEET TACTICAL COMMAND (U)	£39.95
FTC DATA PRINTER	£9.95
FLIGHTDECK (U)	£15.00
FUGITIVE	£9.95
GREYWOLF	£12.50
OPEN GOLF	£12.50
QUESTION MASTER (U)	£10.00
QUIZ MASTER 2 (U)	£12.50
RETURN TO EDEN	£17.50
SOLITUDE (U)	£15.00
SQUIDGY ROUND THE WORLD (U)	£12.50

LABELLING

ADDRESS BOOK&LABEL PRINTER	£15.00
SUPER DISK LABELLER	£10.00

GENEALOGY

GENEALOGIST 3	£60.00
new pointer driven version	
Upgrade 2nd Edition	£33.00
upgrade other versions, ask	
GENEALOGIST 2ND EDITION	£30.00
BUDGET 128K GENEALOGIST (U)	£12.00

GRAPHICS

IMAGE-D (U)	£10.00
IMAGE PROCESSOR	£15.00
LINE DESIGN	£100.00
PAINTER	£25.00
PICTUREMASTER	£15.00
PICTUREMASTER PLUS	£20.00
QRCTAL	£20.00
QUICK MANDELBROT 3 (U)	£15.00
SCREEN COMPRESSION	£10.00
SCREEN SNATCHER (U)	£10.00
SIMPLE VIDEO TITLES	£5.00
TRANS24 (U)	£10.00

HARDWARE

MICRO PROCESS CONTROLLER	
new redesigned version	£64.50
MPC SOFTWARE	£9.95
NETWORK PROVER	£4.00
QPOWER REGULATOR	£24.95
(unsuitable Gold Card)	
SERMOUSE	£40.00
VOICE ANALYSER-New!	
Ask for details and price	
Add £2 50 postage for MPC,	
SERMOUSE AND VOICE ANALYSER	

MAGAZINES

QREVIEW	£2.00
QL TECHNICAL REVIEW	£2.00
Issues 1-8 available	
QL ADVENTURER'S FORUM	£1.75
Issues 1-9 available	
QL LEISURE REVIEW	£2.00
Issues 1-2 available	
(UK prices only shown)	

PROGRAMMING

BASIC REPORTER (U)	£10.00
DISA	£29.00
DUTTOOLKIT (U)	£10.00
EASYPTR 3 PART 1	£40.50
EASYPTR 3 PART 2	£20.00
EASYPTR 3 PART 3	£20.00
MEGATOOLKIT DISK (U)	£25.00
MEGATOOLKIT EPROM (U)	£40.00
QLIBERATOR 3 36	£50.00
BUDGET QLIBERATOR (U)	£25.00
QLOAD AND QREF (U)	£15.00
S EDIT EDITOR	£20.00

SCREEN DUMPS

SIDEWINDER PLUS	£24.95
-----------------	--------

SPREADSHEET PRINTING

SIDEWRITER (U)	£15.00
3D TERRAIN	£12.50

SUNDRIES

3 5" DSDD DISKS each	£0.40
3 5" DSHD DISKS each	£0.70
3 5" DISK LABELS (roll 100)	£2.00
3 5" LABELS (printer roll)	£2.50
ADDRESS LABELS (roll 100)	£2.00
MICRODRIVE CARTRIDGES	£2.50
MDV LABELS (roll 100)	£2.00
MOUSE MAT	£2.50
3 5" DISK DIVIDERS (20)	£3.00
Add £0 50 postage for labels or mouse	
mat if only ordering those, or £2 50 postage	
for disks or disk box divider sets	

TEXT

BANTER BANNERS	£25.00
BIBLE TEXT	£20.00
QTY2	£29.95
QUICK POSTERS (U)	£10.00
ROB ROY PACK (U)	£10.00
SPELLBOUND	£30.00
SPELLBOUND S E	£50.00
TEXTIDY	£15.00
TEXT 'N' GRAPHIX	£20.00

OTHER SOFTWARE

HOME BUDGET (U)	£20.00
PRINTERMASTER (U)	£20.00
QPAC1	£19.95
QPAC2	£39.95
THE Pointer Environment package!	
QTOP	£29.95
SCREEN DAZZLER	£15.00
SCREEN ECONOMISER (U)	£10.00
SLOWGOLD (U)	£5.00
SPEEDSCREEN (U)	£15.00
SPEEDSCREEN EPROM (U)	£30.00
TASKMASTER	£25.00
VISION MIXER 1	£10.00
VISION MIXER PLUS	£22.50

CALL OR WRITE FOR A FREE COPY OF
OUR QL SOFTWARE CATALOGUE
DETAILING AROUND 100 QL PRODUCTS

PLEASE NOTE: (U) ABOVE MEANS THAT
THE SOFTWARE IS SUITABLE FOR USE
ON A 128K QL



TERMS: POSTAGE-software sent post free to UK, overseas add £1 00 per program (maximum £3 00) Floppy disks, SERmouse, etc add £2 50 postage (see above) PAYMENT - IN UK currency (pounds sterling) only, please Valid methods of payment are cheque drawn on UK branch of bank or building society, Eurocheque, Postal Order, cash (send by registered post) or by credit card - Visa, Access, Mastercard and Eurocard accepted Please make cheques etc payable to DILWYN JONES COMPUTING Minimum order value £5 00 (due to bank charges) Goods remain property of Dilwyn Jones Computing until paid for in full Orders normally sent out within a few days of receipt, we will attempt to advise if any delay anticipated due to stock problems Orders can be accepted by telephone if paid for with credit card Orders paid with credit card can only be sent to cardholder's address under card company rules FAX - We now have a Fax machine on our usual number (0248) 354023

